Bringing the Subject Domain Ontology to Optimal Canonical Form

Anatoli Karpuk
Belarusian State Academy of Communications
Minsk, Belarus
Email: a_karpuk@mail.ru

***Abstract*—A formal definition of the subject domain ontology is given. The concept of the canonical form of a subject domain ontology is considered, which is built on the basis of an analysis of functional dependencies between concepts and properties of the subject domain ontology concepts. An algorithm for bringing the subject domain ontology to a canonical form is described. The concept of the optimal canonical form of a subject domain ontology is introduced, containing the minimum number of classes and the minimum number of attributes in the classes. A method for bringing the subject domain ontology to the optimal canonical form is roposed.**

***Keywords*—ontology, domain, canonical form, functional dependence, optimal canonical form**

I. Introduction

In computer science, ontology is a comprehensive and detailed formalization of a certain area of knowledge in the form of a conceptual diagram. A conceptual schema is a set of concepts and information about concepts, which includes properties, relationships, restrictions, axioms and statements about concepts necessary to describe the processes of solving problems in a selected subject domain. For each knowledge area, an applied ontology is built, which consists of a top-level ontology, a subject domain ontology, and a task ontology. Subject domain ontology are simultaneously developed and used by many users. For this reason, in subject domain ontology, the same property of a concept can be represented in different ways. Such ambiguity can lead to difficulties when solving problems using subject domain ontology. To eliminate this drawback, works [1], [2] propose to bring the subject domain ontology to the so-called canonical form, which is based on the analysis of functional dependencies between the concepts and properties of the ontology. However, the developed algorithm for bringing the ontology to a canonical form also does not provide a unique solution. Depending on the order in which the functional dependencies between attributes are analyzed, it is possible to obtain a different number of classes with different numbers of attributes in them. This article introduces the concept of an optimal canonical form of a subject domain ontology, containing a minimum number of classes and a minimum number of attributes in classes, and proposes a method for bringing a subject domain ontology to an optimal canonical form.

II. Formal Definition the Subject Domain Ontology

Let us define the subject domain ontology in the form of a quadruple $O = \langle K, R, F, I\rangle$ [3], [4] where K is a finite set of concepts of the subject domain ontology; R – a finite set of relations between concepts; F – a finite set of interpretation functions defined on concepts and relationships; I – a finite set of axioms, each of which is always a true statement on concepts and relations. The set of concepts has the form $K = \langle D, A, Q\rangle$, where D is a finite set of domains; A — a finite set of attributes; Q — a finite set of classes in subject domain ontology.

Domains are used as sets of possible attribute values. Each domain's data has one of the data types allowed in the XML language [5]. Based on the number of data elements in the value, domains are divided into atomic, union, and list. An atomic domain consists of indivisible data elements of a specific type and format. The value of a federated domain is a data aggregate (structure) consisting of other aggregates and data elements. The value of any merged domain can be represented as a union of the values of its constituent atomic domains. A list domain value is a list (repeating group) of atomic or concatenated domain values. The number of list elements can be any. The value of any list domain can be represented as a repeating group of values from one or more atomic domains.

Based on value restrictions, atomic domains are divided into primitive, built-in, and constructed. Primitive data types of the XML language are used as primitive atomic domains. Built-in domains are derived from primitive domains by applying fixed constraints to them. For example, the primitive domain decimal produces the built-in domains integer, long, int, short, byte, nonNegativeInteger, positiveInteger, unsignedLong, unsignedInt, unsignedShort, unsignedByte, nonPositiveInteger, negativeInteger. Derived domains are derived from primitive and built-in domains by applying various facets to them. For example, the constraints length, minLength, maxLength, pattern, enumeration, whiteSpace, assertions can be applied to the primitive atomic domain string. The

constraints totalDigits, fractionDigits, pattern, whiteSpace, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive, assertions can be applied to the primitive atomic domain decimal.

Depending on the identification method, domains can be unnamed or named. Unnamed domains do not carry semantic load and are used only as data types. Unnamed domains cannot be merged or list domains. Named domains are distinguished from unnamed domains by having a user-defined domain name and can be atomic primitives, inline and derived domains, as well as federated and list domains.

Concepts from set A represent properties (attributes) of subject domain classes. Each attribute is specified by its unique name and the domain to which the attribute values belong. Depending on what domain the attribute is defined on, it can be atomic, aggregated, or list.

Concepts from set Q represent classes (objects, entities) of the subject area. One class includes real or abstract people, objects, phenomena, events, processes that have the same or similar set of properties (attributes), knowledge about which is stored in the ontology and used when solving problems from a given subject domain. When constructing a subject domain ontology, classes are first described, and then knowledge about the individuals of each class is recorded in the ontology. In Russian-language literature, individuals of classes are often called instances of classes or objects. Each class is given its own unique name and its own set of attributes.

Each attribute within a class can have cardinality and functional properties. Cardinality values indicate the minimum and maximum number of attribute values that one individual of a class can have. By default, a class individual can have any number of attribute values. If an attribute value may not be present in an individual of a class, then the ontology must indicate a minimum cardinality equal to 0. The functionality sign shows that any individual of a class can have no more than one attribute value. If the functionality attribute is set, then the maximum cardinality of this attribute should not be specified, or should be equal to 1. The set of class attributes, the values of which uniquely determine an individual of the class, is declared as the key of the class. A class can have more than one key.

The set of relations between concepts R includes relations between domains for constructing derived domains, relations between attributes and domains for determining the scope of attributes, relations between classes and attributes for determining the composition of the attributes of each class, and relations between classes. Relationships between classes reflect "whole-part", "genus-type" connections, as well as hierarchical and other connections between classes that exist in the subject domain. Each relationship between classes can also have cardinality and functionality properties. In addition, relationships between classes can be inverse, inverse functional, transitive, symmetric, asymmetric, reflexive, and irreflexive.

The set of functions F consists of n-ary relations between classes or attributes in which the value of an element with a number n is uniquely determined by the values of previous $(n-1)$ elements. Using functions, you can describe class keys, hierarchical relationships between classes and attributes, and any other functional dependencies between classes and attributes that exist in the subject domain.

The set of axioms I serves to represent in the ontology statements about classes, attributes, domains and relations that are always true. Each axiom is formulated in the form "if <condition on the values of domains for given attributes of given classes or relations> then <statement about the values of domains for given attributes of given classes or relations>". Axioms are included in the ontology to check restrictions on the values of attributes, to check the correctness of the description of the ontology, to derive new true statements about classes, attributes, domains and relationships.

III. Functional Dependencies between Attributes of the Subject Domain Ontology

Let $X \subset A$ be a subset of attributes of the subject domain ontology, $Z \in A$ — some attribute. We will say that in the subject domain ontology there is a functional dependence (FD) $X \to Z$, if any combination of attribute values from X always corresponds to a single value of the attribute Z.

The FD structure on a set of attributes A satisfies Armstrong's axioms [6]:

if $X \subseteq A$ then $A \to X$ (reflexivity axiom);

if $X \to Y$ and $YC \to D$, then $XC \to D$ (axiom of pseudotransitivity).

If there is a FD $X \to Y$, then they say that X functionally determines Y or Y functionally depends on X. From the given axioms, one can derive a number of properties of the FD structure, which in the literature (for example, [7]) are often also called axioms, although it is more accurate to call them rules of inference. The most important are the following inference rules:

if $X \to YC$, then $X \to Y$ and $X \to C$ (decomposition rule), indeed, by the axiom of reflexivity we have $XYC \to Y$ and $XYC \to C$, then by the axiom of pseudotransitivity we obtain $X \to Y$ and $X \to C$;

if $X \to Y$, then $XC \to YC$ (replenishment rule), indeed, by the axiom of reflexivity we have $XYC \to YC$, then by the axiom of pseudotransitivity we obtain $XC \to YC$;

if $X \to Y$ and $X \to C$, then $X \to YC$ (the union rule), indeed, by the completion rule we have $X \to XY$ and $XY \to YC$, then by the axiom of pseudotransitivity we obtain $X \to YC$.

Obviously, the inclusion relation $\subseteq$ determines the FD structure on the set A, which is called the trivial structure of the FD, and the FDs included in it are called trivial FDs. To specify a FD structure that differs from the trivial one, it is necessary to postulate a finite set of FD $F = \{F_j = X_j \to Y_j \mid X_j \subset A, Y_j \subseteq A, j = \overline{1,m}\}$, which in the article [8] was called a system of generators of the FD structure on the set of attributes A. The FD structure, specified by the system of generators F, will be denoted by $S(F)$.

It is obvious that from $Z \in X$ it follows that $X \to Z$. Such an FD, in which the dependent attribute is part of the left side of the FD, is called trivial. In what follows, we will consider only non-trivial FDs between attributes. In the subject domain ontology, the following non-trivial FDs between attributes can be distinguished:

- each functional attribute of a class that is not a subclass of another class, that is not a subordinate attribute of another class attribute, functionally depends on each class key;
- each functional attribute of a subclass that is not a subordinate attribute of another attribute of a subclass is functionally dependent on each subset of attributes obtained by combining each key of the parent class with each key of the subclass;
- each functional attribute of a class that is not a subclass of another class that is a subordinate attribute of another class attribute functionally depends on each subset of attributes obtained by combining each class key with a parent attribute;
- each functional attribute of a subclass, which is a sub-attribute of another attribute of a subclass, functionally depends on each subset of attributes obtained by combining each key of the parent class with each key of the subclass and the parent attribute;
- each functional relationship between classes from the set R specifies the FD of attributes of each key of the parent class from each key of the subordinate class;
- each function from the set F, defined on the attributes of the subject domain ontology, sets the FD of the last attribute of the relation from the previous attributes of this relation;
- each function from the set F, defined on the classes of the subject domain ontology, specifies the FD of the attributes of each key of the last class of the relation from each subset of attributes containing any one key of the previous classes of this relation.

When defining the FD between the attributes of the subject domain ontology, it is possible to write more than one attribute on the right side of the FD, since for the FD between attributes the property of the cluster decomposition of the right side is valid, namely, if $Y_1 \in A$, $Y_2 \in A$ and $Y = Y_1 \cup Y_2$, then the record $X \to Y$ corresponds to the simultaneous presence of FD $X \to Y_1$ and $X \to Y_2$. Moreover, instead of the last two FD, you can write $X \to Y_1Y_2$. Each FD between attributes in the subject domain ontology can be considered as the simplest rule for deriving new knowledge from the knowledge available in the ontology. Indeed, if the values of the attributes of the left side of the FD are known, then either the ontology already contains uniquely corresponding values of the attributes of the right side of the FD, or the values of the attributes of the right side of the FD can be obtained by solving the problem from the ontology of tasks. The input data of this problem are the values of the attributes of the left side of the FD, and the output data are the values of the attributes of the right side of the FD.

Let us single out in the subject domain ontology all FDs between attributes and represent them as a set of FDs $P = \{P_j = X_j \to Y_j \mid X_j \subset A, Y_j \subseteq A, j = \overline{1,m}\}$, which is called the system of forming FD structures on the set of attributes of the ontology. The structure of the FD, given by the system of generators P, will be denoted $S(P)$.

The closure of the set of attributes $X \subset A$ concerning to the structure of FD $S(P)$ is a set $X^+(P) \subseteq A$ such that for any $Y \subseteq A$ from $X \to Y$ follows $Y \subseteq X^+(P)$. In other words, the closure of the set of attributes X includes all the attributes, the values of which can be obtained from the known values of the attributes of set X, using the FD derivation from the set P. The algorithm for constructing the closure $X^+(P)$ consists of the following steps [8].

1) Put $X^+(P) = X$ and $p_j = 0$, $j = \overline{1,m}$.
2) Put $q = 0$ and for each $j = \overline{1,m}$ perform step 3.
3) If $p_j = 0$ and $X_j \subseteq X^+(P)$ then put $X^+(P) = X^+(P) \cup Y_j$, $q = 1$ and $p_j = 1$.
4) If $q = 1$, then go to step 2, otherwise finish the job.

The structures of FD $S(P^1)$ and $S(P^2)$ on the set of attributes A with systems of generators $P^1 = \{X_i^1 \to Y_i^1 \mid X_i^1 \subset A, Y_i^1 \subseteq A, i = \overline{1,m_1}\}$ and $P^2 = \{X_j^2 \to Y_j^2 \mid X_j^2 \subset A, Y_j^2 \subseteq A, j = \overline{1,m_2}\}$, respectively, are called equivalent if for any $X \subset A$ the equality $X^+(P^1) = X^+(P^2)$. In article [9] it is proven that necessary and sufficient conditions for the equivalence of structures FD $S(P^1)$ and $S(P^2)$ on the set of attributes A are the fulfillment of equalities $X_i^{1+}(P^1) = X_i^{1+}(P^2)$ and $X_j^{2+}(P^1) = X_j^{2+}(P^2)$ for all $i = \overline{1,m_1}$, $j = \overline{1,m_2}$. The system of generators $E = \{H_j \to T_j \mid H_j \subset A, T_j \subseteq A, j = \overline{1,m}\}$ is called an elementary basis of the structure of FD $S(E)$ if the removal of any attribute from the left or right side of any FD from E leads to the structure of FD that is not equivalent to $S(E)$.

IV. Canonical Form of the Subject Domain Ontology

We will say that the subject domain ontology is in canonical form if the following conditions are met [1]:

- all attributes from the set A participating in the definition of classes, functions, and axioms ontology are atomic;
- all attributes of each class have a functionality flag and have no subordinate attributes;
- the system of FD structure generators between the attributes of the ontology is the elementary basis of this FD structure.

The algorithm for reducing the subject domain ontology to the canonical form consists of the following steps.

1) For each composite attribute, add to set A the atomic attributes that make up the com-posite attribute. If this composite attribute is part of some class with a flag of functionality, then replace it in this class with atomic attributes with a flag of functionality. If a composite attribute is a part of a class without a functionality flag (it is a list attribute), then represent it as a subclass consisting of atomic attributes with a functionality flag included in the composite attribute. At the same time, determine the keys of a new class depending on the presence of an FD between atomic attributes within a composite attribute.
2) Each composite attribute included in the functions and axioms of the ontology should be replaced with atomic attributes included in its composition.
3) Each atomic attribute that is part of some class without a flag of functionality, to represent in the form of a subclass consisting of this attribute with a flag of functionality, and the class key consists of this atomic attribute.
4) Each atomic attribute that is part of a class and has subordinate attributes in it should be represented as a subclass consisting of this attribute and subordinate attributes with a flag of functionality. If this attribute was without the functionality flag, then the key of the new class consists of this atomic attribute otherwise all of its attributes are included in the key of the new class.
5) Select non-trivial FD between attributes according to the rules described above and form a system of generators of FD structure on the set of attributes $P = \{P_j = X_j \rightarrow Y_j \mid X_j \subset A, Y_j \subseteq A, j = \overline{1,m}\}$.
6) Remove redundant attributes from the left sides of the FD from set P. Attribute $B \in X_j$ is considered redundant in X_j, if $B \in (X_j \backslash B)^+(P)$.
7) Remove redundant attributes from the right sides of the FD from set P. Attribute $B \in Y_j$ is considered redundant in Y_j, if $B \in {X_j}^+(P')$, where P' denotes the system of generators of the FD structure, obtained from P by replacing FD $X_j \rightarrow Y_j$ with $X_j \rightarrow (Y_j \backslash B)$. As a result of performing steps 6 and 7, an elementary basis of the FD structure on a set of attributes $E = \{H_j \rightarrow T_j \mid H_j \subset A, T_j \subseteq A, j = \overline{1,m}\}$ will be obtained.
8) Bring the subject domain ontology in accordance with the obtained elementary basis of the FD structure between attributes by performing the following steps:
   - remove from the classes the attributes that turned out to be redundant in the right parts of the corresponding FD of the elementary basis;
   - remove from the composition of the keys of the classes the attributes that turned out to be redundant in the left parts of the corresponding FD of the elementary basis;
   - unite into one class those ontology classes that have the same closures of their keys concerning the elementary basis of the FD structure;
   - remove functions from the set F, in which all the attributes of the right-hand sides in the corresponding FD of the elementary basis turned out to be redundant;
   - from the left-hand sides of the functions from the set F, remove the attributes that turned out to be redundant in the left-hand sides of the corresponding FD of the elementary basis.

V. Optimal Canonical Form of the Subject Domain Ontology

The canonical form of a subject domain ontology is called optimal if it contains a minimum number of classes with a minimum number of occurrences of attributes in them. The task of bringing the subject domain ontology to optimal canonical form comes down to finding the optimal elementary basis of the structure FD between the attributes of the ontology, which contains the minimum number FD with the minimum number of occurrences of attributes in them.

In article [9], the concept of P-dependencies in the elementary basis of the FD structure was introduced and studied. Let $E = \{H_j \rightarrow T_j \mid H_j \subset A, T_j \subseteq A, j = \overline{1,m}\}$ be the elementary basis of the FD structure $S(E)$ on the set A on the set A. We will say that in the FD $(H_s \rightarrow T_s) \in E$ there is a P-dependence of a non-empty ${T_s}' \subseteq T_s$ on H_s if there exists an $P \subset {H_s}^+(E)$, $P \neq H_s$ such that ${T_s}' \subseteq (P^+(E) \backslash P)$ and no subset of P possesses these properties. Let E' us denote the system of generators of the FD structure obtained from E by replacing the FD $H_s \rightarrow T_s$ with $H_s \rightarrow T_s \backslash {T_s}'$. We will distinguish three types of P-dependence: P_1-dependence occurs if $P \subseteq {H_s}^+(E')$, P_2-dependence if simultaneously $P \nsubseteq {H_s}^+(E')$ and ${T_s}' \nsubseteq (P^+(E' \backslash P)$, P_3-dependence if simultaneously $P \nsubseteq {H_s}^+(E')$ and ${T_s}' \subseteq (P^+(E' \backslash P)$.

We formulate the main properties of P-dependencies in the form of the following statements, the proof of which is carried out by checking the fulfillment of sufficient conditions for the equivalence of the FD structures.

If in the FD $(H_s \to T_s) \in E$ there is a P_1-dependence of $T_s^{'} \subseteq T_s$ on H_s, and the system of generators of the structure of the FD Q is obtained from E by replacing the FD $H_s \to T_s$ with $H_s \to T_s \backslash T_s^{'}$ and adding the FD $P \to T_s^{'}$ to the E, then the structures of the FD specified by the systems of generators E and Q are equivalent.

If in the FD $(H_s \to T_s) \in E$ there is a P_2-dependence of $T_s^{'} \subseteq T_s$ on H_s, and the system of generators of the FD structure Q is obtained from E by replacing the FD $H_s \to T_s$ with $H_s \to ((T_s \backslash T_s^{'}) \cup (P \backslash H_s))$ and adding the FD $P \to T_s^{'}$ to the E, then the FD structures specified by the systems of generators E and Q are equivalent.

If in the FD $(H_s \to T_s) \in E$ there is a P_3-dependence of $T_s^{'} \subseteq T_s$ on H_s, and the system of generators of the FD structure Q is obtained from E by replacing the FD $H_s \to T_s$ with $H_s \to ((T_s \backslash T_s^{'}) \cup (P \backslash H_s))$, then the FD structures specified by the systems of generators E and Q are equivalent.

The given properties of P-dependencies make it possible to move from one elementary basis of the FD structure to other elementary bases and find the optimal elementary basis of the FD structure.

In article [10], the concept of a cycle in the elementary basis of the FD structure is introduced and it is proved that the presence of cycles in the elementary basis of the FD structure is a necessary condition for the existence of P-dependencies in the elementary basis. The elementary basis of the FD structure on set A can be associated with a bipartite oriented graph (A, E, H, T), in which A – the set of vertices of the first part of the graph, E – the set of vertices of the second part of the graph, H – the set of arcs of the graph directed from the vertices of the first part to the vertices of the second part of the graph (showing the occurrence of elements from A to the left parts of the FD), T is a set of arcs of the graph directed from the vertices of the second part to the vertices of the first part of the graph (showing the occurrence of elements from A in the right parts of the FD). It is easy to verify that each cycle in an elementary basis corresponds to a family of cycles in the corresponding bipartite graph.

In general, the problem of finding all cycles in a bipartite directed graph is NP-hard, but its difficulty is determined by the fact that the maximum possible number of cycles in a graph depends exponentially on the dimension of the graph. The search time for one cycle in a directed graph using a standard depth-first search algorithm depends linearly on the dimension of the graph. In real optimization problems of the canonical form of a subject domain ontology, with the number of attributes on the order of 10^3, the number of cycles in the elementary basis of the FD structure does not exceed 10^2, therefore all cycles in the elementary basis of the FD structure can be found in an acceptable time.

VI. Software for Bringing the Subject Domain Ontology to Canonical Form

The input data of the software is the subject domain ontology in the OWL-2 language in the input file. The output data is the equivalent subject domain ontology, which is in the canonical form, presented as an owl-file. The software extracts from the original owl-file and presents attributes, classes, class hierarchy, links between attributes and classes in the form of database tables. Then, atomic attributes and the system of forming the FD structure between the attributes are extracted from the database tables. The software for bringing the subject domain ontology to the canonical form includes the Attribute, AttributeSet, FuncDepen, FDStructure classes developed in C++.

The Attribute class is used to represent a single atomic attribute. The class data is an attribute identification number, an attribute name, and an attribute purpose.

The AttributeSet class is used to represent any subset of atomic attributes. The class data is an array of attribute identification numbers. The class methods return the number of attributes in a subset, add an attribute to a subset, remove an attribute from a subset, check for the presence of an attribute in a subset, check if a given subset of attributes is in a subset, get the union and intersection of a given subset of attributes with a subset.

The FuncDepen class is used to represent a single functional dependency between atomic attributes. The data of the class are an object of the AttributeSet class, corresponding to the left part of the FD, and an object of the AttributeSet class, corresponding to the right part of the FD. Class methods add an attribute to the left or right part of the FD, remove an attribute from the left or right part of the FD.

The FDStructure class is used to represent a system of generators and an elementary basis for a structure of functional dependencies between atomic attributes. The class data is an array of objects of the FuncDepen class. The class methods return the number of FDs in the structure, add FDs to the structure, remove FDs from the structure, obtain the closure of a given subset of attributes with respect to the FD structure, and find the elementary basis of the FD structure.

Software for bringing the subject domain ontology to the optimal canonical form is under development.

The software for bringing the subject domain ontology to canonical form was used in the development of the domain ontology of radio communication networks. As a result, the main ontology classes and their subclasses were obtained.

The TransmiterTypes class and its subclasses are designed to store data about transmitter types. The

classes include attributes: name of the transmitter type, boundaries of operating ranges frequencies, power range boundaries, emission codes, emission bandwidth for each emission code, dependence of the attenuation of out-of-band and noise emissions on detuning from the operating frequency, attenuation of radiation at harmonic and reference oscillator frequencies, attenuation of radiation at combination and intermodulation frequencies.

The ReceiverTypes class and its subclasses are designed to store data about receiver types. The classes include attributes: the name of the receiver type, the boundaries of the operating frequency ranges, sensitivity, codes of received emissions, bandwidth and the required signal-to-noise ratio for each received radiation, the dependence of the sensitivity attenuation on detuning from the operating frequency, intermediate frequencies, the radiation power of the receiver at local oscillator frequencies, weakening of sensitivity at intermediate frequencies, local oscillator frequencies, mirror local oscillator frequencies, combination and intermodulation frequencies.

The AntennaTypes class is designed to store data about antenna types. The class includes the following attributes: antenna type name, antenna type code, polarization code, minimum and maximum electrical center height, isotropic gain for horizontal and vertical polarization, antenna half power beamwidth in horizontal and vertical plane, side lobe attenuation relative to to an isotropic antenna, attenuation in the feeder.

The RadioDevTypes class is designed to store data about types of radio devices (RD). The class includes the following attributes: name of the RD type, code of the RD type (transmitter, receiver, radio station), code of the RD operating mode (simplex, duplex).

The ObjectCommInds class is designed to store data about individuals of communication objects. The class includes the following attributes: name of the communication object, geographic coordinates of the center of the stationary communication object or the center of the movement zone of the mobile communication object, the radius of the movement zone of the mobile communication object, the height of the point of standing of the stationary communication object above sea level.

The AntennaInds class is a subclass of the AntennaTypes class and is intended to store data about antenna individuals. The class includes the following attributes: the name of the antenna, the height of the electrical center of the antenna above the communication object, the direction angles of the antenna in the horizontal and vertical plane, the coordinates of the antenna relative to the center of the communication object.

The Radiolines class and its subclasses are designed to store data about radio lines (RL) and radio networks. The classes include the following attributes: name of RL, RL importance code, RL type code (with fixed radio frequencies, with pseudo-random switching of the operating frequency, radio relay line interval), RL operating mode code (simplex, duplex with time division, duplex with frequency division), codes of emissions used in RL, frequencies or average frequencies of frequency bands assigned to RL for transmitting and receiving in the main RD.

VII. Conclusion

To eliminate ambiguity and redundancy in the domain ontology, the concept of the canonical form of the domain ontology was introduced and algorithms were proposed for bringing the domain ontology to a canonical form and an optimal canonical form. Software has been developed that implements bringing the domain ontology to a canonical form.

References

[1] A. A. Karpuk and A. V. Havorka, “Bringing the Subject Domain Ontology of Radio Communication Networks to Canonical Form”, Problems of Infocommunications, № 2(14), pp. 25–30, 2021, (in Russian).

[2] A. V. Havorka and A. A. Karpuk, “Reduction the Subject Domain Ontology to Canonical Form”, International Journal of Information and Communication Technologies. Special Issue, pp. 43–47, May 2022.

[3] A. V. Palagin, S. L. Kryvy and N. G. Petrenko “Ontological Methods and Means of Processing Subject Knowledge: Monograph”, Lugansk, 324 p., 2012, (in Russian).

[4] A. A. Karpuk and A. V. Havorka, “Construction of an Applied Ontology of Radio Communication Networks”, Vestnik suvjazi, № 6, pp. 36–40, 2021, (in Russian).

[5] “W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes. W3C Recommendation”, 5 April 2012 [Electronic resource] URL: http://www.w3.org/TR/xmlschema11-2.

[6] W. W. Armstrong, “Dependency structure of data base relationships”, Proc. IFIP Congress. Geneva, Switzerland, pp. 580–583, 1974.

[7] S. D. Kuznecov, “Database. Models and Languages”, M., Binom–Press, 720 p., 2008, (in Russian).

[8] A. A. Karpuk and V. V. Krasnoproshin, “Methodology of Data Domain Description for Databases Design in Complex Systems”, International Academy Journal Web of Scholar, Vol. 1, № 4(13), pp. 11–20, 2017.

[9] A. A. Karpuk, “Analysis of Structure of Functional Dependencies between Attributes of a Relational Database”, Economics and Management of Control Systems, № 3(25), pp. 64–70, 2017, (in Russian).

[10] A. A. Karpuk and V. V. Krasnoproshin, “Cycles in Structures of Functional Dependencies”, International Journal of Open Information Technologies, Vol. 5, № 7, pp. 38–44, 2017.

ПРИВЕДЕНИЕ ОНТОЛОГИИ ПРЕДМЕТНОЙ ОБЛАСТИ К ОПТИМАЛЬНОЙ КАНОНИЧЕСКОЙ ФОРМЕ

Карпук А.

Дано формальное определение онтологии предметной области. Рассмотрено понятие канонической формы онтологии предметной области, которая строится на основе анализа функциональных зависимостей между понятиями и свойствами понятий онтологии. Описан алгоритм приведения онтологии предметной области к канонической форме. Введено понятие оптимальной канонической формы онтологии предметной области, содержащей минимальное количество классов и минимальное количество атрибутов в классах. Предложен метод приведения онтологии предметной области к оптимальной канонической форме.

Received 13.03.2024