Обґрунтовано принципову можливість використання онтологічного підходу в організації інформаційної підтримки керівників підприємств авіаційного профілю. Для опису зв'язків між елементами окремих онтологій використаний математичний апарат операційної семантики. Отримані результати передбачається використовувати в якості методичної основи для створення машини виведення як елемента ядра виробничих систем підтримки прийняття рішень керівниками підприємств авіаційного профілю

Ключові слова: підприємства авіаційного профілю, мова операційної семантики, система підтримки прийняття рішень, онтологічний інжиніринг

Обоснована принципиальная возможность использования онтологического подхода в организации информационной поддержки руководителей предприятий авиационного профиля. Для описания связей между элементами отдельных онтологий использован математический аппарат операционной семантики. Полученные результаты предполагается использовать в качестве методической основы для создания машины вывода как элемента ядра производственных систем поддержки принятия решений руководителями предприятий авиационного профиля

Ключевые слова: предприятия авиационного профиля, язык операционной семантики, система поддержки принятия решений, онтологический инжиниринг

UDC 004.942
DOI: 10.15587/1729-4061.2018.126673

ORGANIZATION OF INFORMATION SUPPORT FOR BUSINESS PROCESSES AT AVIATION ENTERPRISES BY MEANS OF ONTOLOGICAL ENGINEERING

I. Shostak
Doctor of Technical Sciences, Professor
Department of Software Engineering*
M. Danova
PhD
Department of Software Engineering*
Yu. Romanenkov
Doctor of Technical Sciences, Associate Professor
Department of Management*
E-mail: KhAI.management@ukr.net
O. Bugaienko
PhD
Department of Chemistry, Ecology and Expert Technologies*
M. Volk
PhD, Associate Professor
Electronic Computers Department
Kharkiv National University of Radio Electronics
Nauki ave., 14, Kharkiv, Ukraine, 61122
M. Karminska-Bielobrova
PhD
Department of production organization and personnel management
National Technical University "Kharkiv Polytechnic Institute"
Kyrpychova str., 2, Kharkiv, Ukraine, 61002
*N. E. Zhukovsky National Aerospace University "Kharkiv Aviation Institute"
Chkalova str., 17, Kharkiv, Ukraine, 61070

1. Introduction

At present, information technologies can be used effectively to solve the tasks on creation, transmission, processing, and storage of production and management information [1]. Their implementation into actual process of production and management made it possible to create technological, telecommunication and professional infrastructures of an enterprise for further qualitative step in the deployment of knowledge information technologies – creation of a comprehensive enterprise management system based on the knowledge technology. Integration of expertise of all employees in order to solve management and production tasks underlies the development and support of competitiveness of an enterprise [2]. Application of network information technologies based on the principles of joint use of personnel intelligence and capabilities of expert and analytical systems opens up new opportunities for the integrated solution to these problems. Thus, it is possible to argue that modern network technologies, based on the application of specialized software, provide experts-analysts with a possibility to collect, process and analyze large amounts of information, which would be inaccessible if traditional management technologies were employed.

The specificity of aviation enterprises management involves the need to take into consideration a frequent modification of products, the existence of a great number of relationships between production facilities, small scale production, additional stages of quality control, the need to ensure high levels of reliability, as well as precise control over

technological processes. Given this, it should be noted that intellectualization is an effective way of automation. This process is implemented by creating subject-oriented expert systems (ES) [3]. That is predetermined by the fact that the traditional approach to the synthesis of ES and construction of knowledge bases for a given type of production and the production control level will not enable an administrator to manage a knowledge base effectively, for example, a production type base. In addition, to ensure the systems properties of an object (controllability, observability, stability), it is necessary to attract specific methodological tools for linking different components of the system into a unified information space [4].

Knowledge management in a unified information production environment is currently one of the main factors for achieving flexibility and dynamism in the aviation production [5]. In this case, it is possible to reduce the time needed for product development using new information technologies that greatly facilitate human labor and automate a production decision making process through the accumulation of experts' experience in a knowledge base in order to subsequently apply it in ES [5]. Given the aforementioned, the main tasks for the optimal real-time control over a distributed decision support system (DSS) are: rational organization of production, processing and storage of the knowledge base (KB) of intelligent processes at an enterprise [6]. The main feature of such information technologies [7] is partial intellectualization of decision support processes at various levels of the generalized process of control over complex, dynamic, heterogeneous, and distributed, in the general case, objects, which operate in a changing competitive environment.

At the same time, traditional technology for the creation and deployment of ES includes a number of structural flaws that hamper the effective use of such systems in a production environment [8]. First, empty shells of ES, based on which expert systems are mostly created, because of their universality, do not make it possible to take into consideration the peculiarities and specificity of certain production managers. Second, the use of such shells requires hard work on creating KB for every user of production ES. This fact inhibits complex automation of production with the use of ES [9, 10]. Third, ES by their very design are closed systems, which is inconsistent with the concept of creating a unified information space of production. The above problems can be effectively solved by using the models of ontological knowledge as an environment for building knowledge bases of production ES [9–11]. To develop a distributed intelligent decision support system (DIDSS) based on the ontological approach, it is possible to use both modern high-level languages such as C#, C++, VB.NET from the very beginning, and existing systems like Protégé, CLIPS [12]. However, this can lead to constraints for the functional content of a system due to its internal features, such as using only direct inference on knowledge.

Thus, under present-day conditions, it is an important scientific and applied task to design ontological methods for creating and operating production decision support systems, in particular at aviation enterprises. Solving this task would make it possible to improve efficiency of production automation processes, as well as effectiveness of managerial decision support processes at aviation enterprises, by employing an ontological knowledge system as the ES core.

2. Literature review and problem statement

Paper [9] showed that effectiveness of solving complex applied problems, related to organization of information support of business processes at aviation enterprises, largely depends on objectivity and reliability of the information, used throughout the whole process of their solution. It addition, it should be borne in mind that objects (concepts), the features of which determine conditions and stages of solving most of applied problems can belong to different subject domains. Authors of this article conducted a study of the need for integration of the used information and data that characterize it, based of thematic properties of information units, determining the chosen solution strategy [10, 13–17].

The above-mentioned papers contain the results of application of ontological engineering in various subject domains. Thus, paper [10] considered the issues of creating, deploying, and maintaining a unified knowledge space in intelligent integrated systems at industrial enterprises. Publications [13, 14] address the problems of application of ontological approach for tasks on scientific and technological prediction. Papers [15, 17] contain descriptions of theoretical and applied results of the application of ontological engineering for the problems, related to software development, as well as manufacturing highly technological products [16]. This, in turn, defines a number of tasks on scientific, methodological and technological support of an organization and work with information, specifically: effective tools and means of accumulation and storage, search and analysis, mining and using a knowledge about organization of production at aviation enterprises. In this case, information arrays have a relatively large amount, which results in a number of additional problems.

Today, the synthesis of knowledge representation models [18] for solving decision support problems is one of the most dynamic directions in information technologies. In this case, the main problem in this area is to search for rational forms of knowledge representation for storing in computer memory. These forms of knowledge, referred to as knowledge representation models, must be sufficiently expressive, easy to use, and effective for manipulation [19]. Expressiveness implies the ability of the model to represent equally well both generalized and specific knowledge from universal and special subject domains. The specified property also determines the possibility to display incomplete knowledge. Usability means that the model for knowledge representation should be based on the concepts, used by experts and users. The knowledge model should be formalized, because by DSS knowledge we imply formalized data, referred to in the inference process [20].

Today, the most widely used knowledge representation models are: a logical model, a production model, a model in the form of semantic network, a frame model, an ontological model [21–23]. The knowledge representation languages (KRL) are used for development of DSS, based on the specified models [24].

Taking into consideration the specificity and the subject domain environment, specifically small scale, frequent modification changes, increased requirements to quality control, high technological effectiveness of production, we will consider publications that address the synthesis of the following knowledge representation models: semantic networks, scenarios, frames, and ontological models.

Traditionally, it is common to describe KB as a representation of objects and relations in a subject domain into computational objects and program relationships [22]. Inference results in KB must comply with the results of action or observations in the world. Computational objects, relations, and conclusions, habitual for a programmer, are expressed by KPL [25].

There are general principles of knowledge organization, which are applied in various fields and can be directly supported by KRL. For example, class hierarchies [26] can be found in both scientific and natural classification systems. However, the problem of creation of a common mechanism for representation and axiomatization, DSS learning, decreasing uncertainty level and integrity check depends on the application of a number of special principles of knowledge organization and on their support by high-level representation tools. KRL are usually more limited than calculation of predicates or programming languages. These constraints take the form of precise structures for representation of knowledge models. The environment, in which they are performed, can be PROLOG, LISP or Microsoft.NET environment [26].

These issues were addressed by studies into the area of robotics [27, 28], situational cognition [1] and agent solution of problems [29]. The above papers challenged the traditional approach to knowledge representation. These problem areas require distributed knowledge; the world, which itself can be used as a particular knowledge structure; possibility to reason at incomplete information based on representations that vary in the process of experiments in the problem area. These principles are especially characteristic for the ES, oriented to solution of applied problems.

One of the major challenges arising during ES construction is the development of formal knowledge representation language with perfect and complete inference rules [30]. There are many issues that arise when displaying judgments based on common sense into the formal logic, which prevents direct use of these judgments in ES. Article [31] showed that the modern level of technologies of constructing complex information systems is characterized by modularity of project construction, attained by algorithmic or object-oriented (OO) decomposition. The consequence of this desire is the hierarchy of a project, the separation of which is a nontrivial task [13, 16], and only if the abstraction level lies within the 7+/–2 [10], the congruent system control in general is possible. Restriction in the 7+/–2 abstraction level, makes it impossible to use standard technologies for construction of highly developed ES.

Associationist theories determine the value of an object in terms of a network of associations with other objects [32]. From the stand point of this theory, perception of an object is performed through concepts. Concepts are a part of all our knowledge about the world and are linked to other concepts by correspondent associations. These links represent the properties and behavior of an object.

Inheritance systems allow us to memorize information at the highest abstraction level, which reduces the DB volume and helps avoid contradictions. The KB volume also decreases due to the requirement to determine substantial properties only once and not to determine them for each object. In addition, inheritance helps to maintain KB consistency when adding new classes and an object.

For formalization of associationist knowledge theories, the graphs proved to be an ideal tool due to accurate representation of relations through arcs and nodes [33]. The semantic network represents knowledge in the form of a graph, the nodes of which correspond to facts or concepts, and arcs correspond to relationships or associations between concepts [25].

The first computer implementations of semantic networks were created in the early 1960s for the use in automatic (machine) translation systems [25]. Though this and other early works showed the expressive power of graphs for modeling associative sense, they were limited by excessive generality of formalism. Knowledge was usually structured in terms of specific relations, such as object–property, class–subclass and agent–verb–object [25].

The majority of works on network representations were reduced to determining a set of link labels for more comprehensive modeling of the natural language semantics [34].

The papers, devoted to scenarios [25, 35] are important for the development of the conceptual dependence theory [25, 35]. The works on scenarios are associated with studying the knowledge organization in memory and its role in reasoning. Scenario, in this case, is a structured representation, describing a stereotypical sequence of events in a particular context. Scenarios were originally proposed in [36] as a tool for the organization of structures of conceptual dependence in descriptions of typical situations.

A scenario includes the following components [35]: initial conditions that must be true when a scenario is invoked; results or facts that are true, when a scenario is completed; assumptions that support the context of a scenario. Many assumptions describe default scenario conditions. The elements of a scenario, the main "parts" of semantic meaning, are represented as relationships of conceptual dependence. Gathered together in a frame-like structure, they represent a sequence of values or events. Thus, the success of the application of scenarios depends directly on subject domain constraints, not inherent to aviation production.

The most suitable environment for ES is ontological knowledge models that will make it possible to use jointly the benefits of application of ES models and traditional models for ES creation. Theoretical and practical results that currently exist in the area of development of ontological knowledge models, cover [15, 16] mainly only one stage of the knowledge-oriented information technology, specifically, knowledge representation [10, 16, 39]. However, for creation of an ontological knowledge system, which would become the core of the production ES, it is necessary to implement the stage of knowledge manipulation directly in the environment of such a model.

Contemporary ontological systems have a significant disadvantage, which is impossibility of knowledge manipulation directly in the environment of ontological models [16, 38]. Thus, the problem of organization of inference on knowledge within an ontological system is relevant. The solution of the stated problem will make it possible to enhance efficiency of business processes at aviation enterprises due to development and implementation of intelligent decision support systems with the core in the form of a set of ontological models.

To address this issue, it is advisable to use the mathematical apparatus, providing the translation of constructions of the knowledge representation language into the theoretic-multiple model, on which inference semantics is determined, forming a two-level model of formal semantics. This enables the strict description of the mechanism of inter-

action of an ontological model in the distributed information environment, and the actual inference process in it. This approach will make it possible to solve the following problems:

a) to obtain strict mathematical proofs of finiteness of inference in an ontological system through the use of the apparatus of formal semantics;

b) to overcome the difficulty of formalizing implicit rules, relationships and usual perception by formalizing the rules that defy description using language OWL (Web Ontology Language [40]);

c) to carry out verification operations in a strictly formalized ontological model [40], as well as to ensure internal and external knowledge coherence and, ultimately, to ensure the KB completeness;

d) to produce a compilation of the information, represented in the knowledge representation language (KRL), into the internal representation. In this case, the language of formal semantics [25] can be used as a basis for implementation of the correspondent software prototype;

e) to ensure unambiguity and strictness of the formal description of KRL [41], which will make it possible to clearly and strictly represent the syntax and semantics of the language;

f) to implement the problems of control, filling, search, provision of data and knowledge for re-using [31]. These problems arise due to an increase in the number of axioms, rules, concepts, and ontological relations.

3. The aim and objectives of the study

The aim of present research is to create the methods of synthesis of industrial expert systems in the form of intelligent decision support systems based on an ontological knowledge model, which will make it possible to enhance efficiency of the information support on organization of business processes at aviation enterprises.

To accomplish the aim, the following tasks have been set:

– to develop the method for knowledge manipulation in the environment of an ontological model for deductive decision making in production expert systems in the form of intelligent decision support systems;

– to develop a combined method of knowledge manipulation in the environment of an ontological model by combining inductive and deductive ways of decision making in production ES, in the form of intelligent decision support systems.

4. Materials and methods of research

Any subject domain can be described by knowledge KNOWELEDGE, consisting of the non-changing component KWG_{const} and the dynamic component KWG_{dyn}. KWG_{const} is considered as a graph of possible states, and KWG_{dyn} is considered as the rules of transitions by the state graph. In this interpretation of the intelligent system of the production type, KWG_{dyn} describes the knowledge base (KB) as a set of production rules, KWG_{const} describes operating memory, which describes the current stage of problem solving.

To represent knowledge in the form of the ontological models, it is expedient to consider the static component as a set of concepts T, obtained in the process of taxonomic decomposition. This is one of the early stages of development of an ontological model, preceded by the stage of separation of the purpose of an ontological model, definition region (subject domain boundaries) and value region [37]. After that, information about a subject domain is collected and the concepts of a subject domain are separated, it is the so-called stage of taxonomic decomposition.

A set of concepts T is related by the relationship of the form:

– is_a;

– part_of;

– :(inheritance).

In most IDSS, during the inference engine operation, many concepts and their relations remains constant ($KWG_{framework}$), but not the values of fields of concepts, representing operating memory of inference engine KWG_T.

Let us describe the static component of the ontological model IDSS for knowledge representation. In this case, the knowledge that constitutes the core of the $KWG_{framework}$ system is determined by a set of the values of the fields of all KWG_T concepts.

Definition 1. A dictionary of an ontological model, extended by the types and ranges of values from a certain system of types X, is a complete lattice.

Ontology dictionary X can be represented by linearly ordered domains, which are a set of potentially permissible values of this type.

We will use the following formal description of an ontological structure:

$$\Phi : I \to T, \tag{1}$$

where I is the non-extended ontology dictionary; T is the set of concepts; Φ is the function of an ontological structure.

In this case, a display of a set of identifiers I into set of concepts T is such that

$$\forall \tau \in T \exists I_s \left| \tau : I_s \to S, \right. \tag{2}$$

where S is the set of fields.

Representation of a set of concept field identifiers into the set of fields S, unequivocally determines the essence (concept), which in turn allows making operations with set $S \equiv expr(X)$, representing a set of λ-expressions above the X type system:

$$expr(X) \subset \Lambda(X), \tag{3}$$

including relations between concepts, references to other fields of concepts, as well as references to other concepts.

To obtain a unified model of ontological knowledge representation, which includes both static and dynamic component, let us consider the concept field. It is a tuple that consists of the current field value, default value, procedures-daemons, procedures-requests, constraints, etc. Thus, function of state Φ includes a static component (concepts) and the rules of change of this state.

This approach allows determining semantics with dynamic modification of a set of inference rules, constraints, etc. We will formalize the concept field as follows:

$$S = \{(val, def, \{D_i\}, \{Tr_j\}, \{C_k\}, \subseteq_q, \subseteq_d, \alpha, \varsigma)\}, \tag{4}$$

where $val \in X$ is the current value of the field or NULL in case if the value is not determined; $def \in X$ is the field de-

fault value or NULL in case the value is not determined; $\{D_i\}$ is the set of procedures-daemons, joined to the field (each procedure can be an arbitrary expression from a set of expression E^{set}, the structure of which will be considered in more detail below); $\{Tr_j\}$ is the set of the procedures-triggers, joined to the field; $\{C_k\}$ is the set of constraints to the field value, stated in the form of an expression-predicate $C_k \in E^{set}$; $\subseteq_d, \subseteq_{Tr}$ are the linear orders on sets $\{D_i\}$ and $\{Tr_j\}$ respectively, determining the order of application of correspondent inference procedures; $\alpha \in \{true,false\}$ is the flag, pointing out participation of the field in the process of the recurrent bottom-up inference and serving for prevention of infinite loop; $\varsigma \in \{true,false\}$ is the flag, pointing out the field type and determining the class or a class instance, described by this tuple. A trigger starts when a certain condition is met, for example, a value is assigned to a concept field. In accordance with the semantics, described below, procedures-triggers $Tr_j : E^{set} \times \Phi^{set} \to \Phi^{set}$ are applied to the function of state $\Phi \in \Phi^{set}$ in case a certain expression is true and thus generate a new state of the system.

For access to elements

$$S = \{(val,def,\{D_i\},\{Tr_j\},\{C_k\},\subseteq_q,\subseteq_d,\alpha,\varsigma)\},$$

we will use the designations s.value, s.defaultvalue, s.daemons, s.triggers, s.constraints, s.dsequence, s.triggerssequence, s.busy and s.type, respectively. Similarly, we will designate $\tau.s = \langle \tau, s\rangle \in I^{set}, \tau \in I, s \in I_\tau$ for naming a concept, and for the access to the field, we will use $\Phi.\tau.s = \Phi(\tau, s)$.

For further constructions, we will need to introduce the function of field value naming write: $I^{set} \times X \to \Phi^{set} \to \Phi^{set}$, forming a new state, which we will designate as

$$\Phi[\tau, s \leftarrow v] = write(\Phi, \langle \tau, s\rangle, v).$$

This function can be determined as follows:

$$\Phi[\tau, s \leftarrow x] = \lambda\tau_1 \lambda s_1.(\langle \tau_1, s_1\rangle = \\ = \langle \tau, s\rangle \to \Phi(\tau, s)[1 \leftarrow x], \Phi(\tau_1, s_1)), \quad (5)$$

where $(b \to u,v)$ is the operation of conditional computation, and $s[n \leftarrow x]$ designates the function of replacement of the *n*-th component of tuple s with x:

$$(s[n \leftarrow x])_i = (i = n) \to x, s_i.$$

Similarly, we will determine operation

$$\Phi[\tau.s.busy \leftarrow true/false]$$

for assigning a logical value to component s.busy.

We will also determine the operation of difference between states

$$|-| : \Phi^{set} \times \Phi^{set} \to P(I^{set}),$$

returning a list of identifiers of fields, the values of which differ in two given states:

$$\forall \langle \tau, s\rangle \in I^{set} \quad \langle \tau, s\rangle \in |\Phi_1 - \Phi_2| \Leftrightarrow \\ \Leftrightarrow \Phi_1(\tau, s) \neq \Phi_2(\tau, s). \quad (6)$$

State function Φ describes not only the current state in the inference process, but also a set of rules, which in this work is considered constant, as it is not changed in the inference process. Thus, it is possible to separate the function

$$\Phi : I^{set} \to X : \Phi(\tau, s) = \Phi(\tau, s).value \quad \forall \tau \in I,\ s \in I_\tau,$$

which will characterize a purely static component of the system state, as well as the function that is complementary to it, which in turn will be constant in the inference process.

The set of states of the system Ξ can be represented in the form of an infinite graph, the vertices of which will be various states $\Phi \in \Xi$, and the arcs will be assigned by inference rules. Infiniteness of the graph will be primarily caused by potential infinity (in theory – continuity) of the set of values (X) for each of the fields. However, in practice, the number of distinguishable states in each knowledge base will be finite, as there is the finite number of comparisons in the set of references of all rules of the base.

For formalization of this concept, we will introduce for consideration the equivalence ratio ≅, at which

$$\Phi_1 \cong \Phi_2 \Leftrightarrow \Phi_1, \Phi_2$$

are non-distinguishable from the standpoint of the knowledge base, i. e. for the entire set of references C in the left parts of rules of knowledge base $\|c\|_{\Phi_1} = \|c\|_{\Phi_2} \quad \forall c \in C$, where $\|c\|_\Phi$ designates the value of references c in state Φ. Then the space-factor $\Xi_{\cong} = \Xi/\cong$ of space Ξ relative to ratio ≅ will have the following properties:

Statement 1. Space-factor $\Xi_{\cong}$ contains a finite number of elements.

Proof. Space-factor $\Xi_{\cong}$ can be constructed in the following way. Let $c \in C$ be the reference of a certain rule, which can take a true or a false value on state Φ. Then it breaks down a set of states into two sub-sets:

$$\Xi_t(c) = \{\Phi_1 \in \Xi \,|\, \|c\|_{\Phi_1} = true\}$$

and

$$\Xi_f(c) = \{\Phi_2 \in \Xi \,|\, \|c\|_{\Phi_2} = false\}.$$

Let the set of all references C (finite) have the form $C = \{c_1,...c_n\}$. We will construct factor-set $\Xi_{\cong}$ as follows: assume $\Xi^{(1)}_{\cong} = \{\Xi_f(c_1), \Xi_t(c_1)\}$. Then we will determine:

$$\Xi^{(k)}_{\cong} = \bigcup_{x \in \Xi_{\cong}^{(k-1)}} [(x \cap \Xi_f(c_k)) \cup (x \cap \Xi_t(c_k))]. \quad (7)$$

Then $\Xi_{\cong} = \Xi_{\cong}^{(n)}$ will be the sought-for factor-space. Actually, for arbitrary set $A \in \Xi_{\cong}$, if $\Phi_1, \Phi_2 \in A$, then by construction $\forall c \in C \ \|c\|_{\Phi_1} = \|c\|_{\Phi_2}$, i. e. $\Phi_1 \cong \Phi_2$.

Obviously, each transition in space Ξ, assigned by the rule, induces a correspondent transition in space $\Xi_{\cong}$. Thus, exploration of the inference process in the system can be reduced to the study of the finite graph of states with a finite set of transitions.

It is possible to determine the natural order relationship $\subseteq$ on the set of states Ξ as follows:

$$\Phi_1 \subseteq \Phi_2 \Leftrightarrow \forall \langle \tau, s\rangle \in I^{set}\ \Phi_1(\tau, s) \subseteq \Phi_2(\tau, s), \quad (8)$$

where in the second case, the order relationship connects the elements of a set of the type X system. It is easy to make sure that this determining really assigns the order

relationship using the reasonings that are standard for the lattice theory.

It is obvious that if the meta-rules, modifying the dynamic part of the knowledge base in the inference process, are not considered, the order relationship, assigned in this way, also induces the order relationship on set Φ^{set}.

The order relationship, introduced in the formula (8), requires constancy of set I^{set} during inferencing, that is, the names of all concepts and fields must be known in advance, prior to inferencing. These constraints surely narrow down the class of the considered dynamic intelligent systems, and learning a richer semantics is of considerable interest for subsequent research.

In an ontological intellectual decision support system (IDSS), the inheritance relations «:» between concepts is introduced, determining the hierarchy of the subject domain concepts and accordingly used for borrowing properties (fields) of rules from the parent classes.

We differentiate between single and multiple inheritance: in the first case, relation «:» is the function of the right argument, i. e., only one parent can correspond to only child concept, while there is no such constraint in the second case. In the single inheritance, the hierarchy is transformed into a tree.

In addition, there is static inheritance (used, for example, in formalism of F-logic [38]), in which the inheritance relation remains constant in the inference process, and dynamic inheritance, the inheritance hierarchy configuration changes in the inference process.

We will consider the inheritance relationship, induced by values of *parent* fields of child concepts, i. e.

$$T:G \Leftrightarrow \|T.parent\| = G,$$

where $\|T.parent\|$ designates the operation of calculation of a field value, which will be subsequently specified in accordance with the defined semantics. Moreover, to permit the value of the *parent* filed of a certain list type, it is possible to determine the relationship of multiple inheritance as $T:G \Leftrightarrow \|T.parent\| \ni G$.

The inheritance relationship, determined in this way, will be dynamic, because the field value will be calculated in accordance with the rules of semantics, defined below, i.e., it will be possible to determine the parent by means of production rules.

5. Results of research into ontological dependences in subject domain description

The main result, obtained in the course of the study, is the formalization in the language of operating semantics of the process of inference on knowledge in the ontological system environment.

For determining semantics of the inference process, similarly to [42], in the system, we used representation

$$E: E^{set} \times \Phi^{set} \times C^{set} \to X \times \Phi^{set},$$

which calculates the value of arbitrary expression $E \in E^{set}$ in a certain state Φ and in contest C^{set} and returns obtained value $v \in X$ and new state Φ'. We will also use a more convenient designation

$$v = \|E\|^{C}_{\Phi \to \Phi'} \Leftrightarrow E(E,\Phi,C) = (v,\Phi'). \quad (9)$$

Inferential semantics implies that during calculation of the expression value, the inference of values of the fields, for which the value was not known beforehand and was not obtained before in the inference process, can be initiated. Relationship

$$E': E^{set} \times \Phi^{set} \times C^{set} \to X \times \Phi^{set},$$

analogous to E, but not initiating the inference process, was determined. For this display, the following designation was used

$$v = \|E'\|^{C}_{\Phi \to \Phi'} \Leftrightarrow E'(E,\Phi,C) = (v,\Phi').$$

To describe the inheritance semantics in an ontological system, the concept of the calculation context was introduced. The introduction of this concept ensured correctness of application of the rule for the concept-parent to the values in child concepts. It will be enough to put $C^{set}=I^{set}$, although it may be required to include other constructions to the notion of context in more complex models.

The introduction of the basic concept (conveyed through the context) was necessary for calculation of an expression instead of the reserved identifier *this*. In most cases during description of expression calculation semantics, the value of the basic concept is conveyed in the sub-expression calculation function without changes. Thus, if the calculation context is not specified in certain equity, it is implied that the same context appears in both parts of equity.

To determine the semantics, it is necessary to assign the formal syntax for a set of expressions E^{set}. The Backus-Naur form (BNF) or the grammar description was used for the formal syntax assignment. A formal description of the syntax of expression structure $E \in E^{set}$:

– a constant from the set of X types;

– reference to the field in the form of

$$\tau.s,\ \tau \in I \cup \{this\},\ s \in I_{\tau};$$

– an arithmetic or logical operation of the form $E_1 \otimes E_2$, where $\otimes \in \{+,-,*,/,or,and,\to\}$ or negation *not E*;

– operation of calling the function-oracle $A(d),\ d \in D^{set}$.

Arithmetic and logical operations in the set of expressions E^{set} are introduced in accordance with the operations, determined on the set of types. The expression calculation semantics, given in the next sub-section, matches each operation in the syntactic set E^{set} with certain denotate based on determining the operation in set X.

Clearly, interpretation of constants does not depend on state Φ, and its value, i. e. $\forall x \in X\ \|x\|_{\Phi \to \Phi'} = x.$ returns as the denotate of a constant.

The value of expression $E = E_1 \otimes E_2$, which is the arithmetic operation on a pair of expressions E_1 and E_2, will be the corresponding arithmetic operation $\otimes_X$ in a set of types X:

$$\|E_1 \otimes E_2\|_{\Phi_1 \to \Phi_2} = \begin{cases} NULL, & \|E_1\|_{\Phi_1 \to \Phi_2} = NULL, \\ \|E_1\|_{\Phi_1 \to \Phi'} \otimes_X \|E_2\|_{\Phi' \to \Phi_2}. \end{cases} \quad (10)$$

This description takes into consideration the so-called *shortened* expression calculation, in which in case the first

sub-expression returns NULL, the value of the second is not calculated. A simpler approach, when two sub-expressions are calculated in any case, is also possible:

$$\|\mathrm{E}_1 \otimes \mathrm{E}_2\|_{\Phi_1 \to \Phi_2} = \|\mathrm{E}_1\|_{\Phi_1 \to \Phi'} \otimes_{\mathrm{X}} \|\mathrm{E}_2\|_{\Phi' \to \Phi_2}. \quad (11)$$

Computation of logical expressions is carried out similarly, since in the dynamic system of types, clear distinction by the logical and other types is not performed. During calculation of logical expressions, the shortened system is always used. Similarly, the conditional expression is calculated by the shortened scheme:

$$\|\mathrm{E} \to \mathrm{E}_1, \mathrm{E}_2\|_{\Phi_1 \to \Phi_2} = \begin{cases} \|\mathrm{E}_1\|_{\Phi' \to \Phi_2}, isTrue(\|\mathrm{E}\|_{\Phi_1 \to \Phi'}), \\ \|\mathrm{E}_2\|_{\Phi' \to \Phi_2}, not(isTrue(\|\mathrm{E}\|_{\Phi_1 \to \Phi'})). \end{cases} \quad (12)$$

If expression E has the form τ.s, for obtaining its value, it is required to derive the value of the correspondent field from state Φ applying bottom-up inference. Interpretation with the help of function E' does not initiate bottom-up inference, but only returns the value of a slot or NULL (in this case, the state does not change):

$$\|\tau.s\|'_{\Phi_1 \to \Phi_2} = \Phi(\tau.s).\mathrm{value}. \quad (13)$$

Application of bottom-up inference for calculation of field value s involves a sequential (in accordance with a linear order $\subseteq_q$) application of rules from s.rules. Application of the rules lies in calculation of a conditional or unconditional expression, corresponding to this rule, in accordance with Table 1.

Table 1

Correspondence of calculated expression to production rule

No.	Production rule	Expression
1	IF E THEN t.s=T	E→T
2	SET t.s=E	E
3	ASK q t.s	A(q)

Tuple $\langle Q, \subseteq \rangle$ is declared a finite linearly ordered set. We will designate $\langle Q^- = Q \setminus \{\inf Q\}$. Let us introduce function $\mu_{\Phi \to \Phi'}(Q)$ of sequential calculation of the orderly family of expressions Q in the initial state Φ:

$$\mu_{\Phi \to \Phi'}(Q) = \begin{cases} \|\inf Q\|_{\Phi \to \Phi'}, \|\inf Q\|_{\Phi \to \Phi'} \neq NULL; \\ \mu_{\Phi'' \to \Phi'}(Q^-), \|\inf Q\|_{\Phi \to \Phi''} = NULL; \\ NULL,\ \Phi' = \Phi, Q = \varnothing. \end{cases} \quad (14)$$

This recursive definition correctly determined function μ, which is determined for any initial state and finite set Q.

It should be noted that P (Q) is a lattice relative to the inclusion operation, and the order relationship is induced on set Φ^{set} in such a way that it is a lattice, on which function $\|.\|_{\to}$ will be monotonous. In accordance with this, there will be a fixed point $\mu' = fixM$, on display M that will determine the desired function μ'.

In this case, function μ' itself will be monotonous on the lattice, generated by $Q^{set} \times \Phi^{set}$ with considered above order relationships, and according to the Tarsky theorem about a fixed point, it well determine the finite state

$$\langle Q', \Phi' \rangle = \coprod_{n=1}^{\infty} \mu'^n(Q, \Phi).$$

It should be noted that due to finiteness of original set Q and a strict character of monotony, this equality will be satisfied for a certain finite N, i. e.

$$\forall Q \in \mathrm{Q}^{\mathrm{set}}\ \exists \mathrm{N} \leq \#Q:\ \langle Q', \Phi' \rangle = \coprod_{n=1}^{N} \mu'^n(Q, \Phi). \quad (15)$$

Using function μ, determined in this way, function $\|.\|$ for the field, taking into consideration bottom-up inference, is determined:

$$\|\tau.s\|_{\Phi \to \Phi'} = \begin{cases} \Phi(\tau.s).\mathrm{value},\ \Phi(\tau.s).\mathrm{value} \neq NULL\ (\Phi' = \Phi); \\ \mathrm{NULL},\ \Phi(\tau.s).\mathrm{busy=true}; \\ x = \mu_{\Phi[\tau.s.busy \leftarrow true] \to \Phi''}(\langle \Phi(\tau, s).\mathrm{rules},\ \Phi(\tau, s).\subseteq_q \rangle), \\ \Phi' = \Phi''[\tau.s \leftarrow x,\ \tau.s.\mathrm{busy} \leftarrow \mathrm{false}]\ (x \neq \mathrm{NULL}); \\ y = \mu_{\Phi''' \to \Phi''''}(\{p.s \mid p \in \|\tau.\mathrm{parent}\|_{\Phi'' \to \Phi'''}\}), \\ \Phi' = \Phi''''[\tau.s \leftarrow y,\ \tau.s.\mathrm{busy} \neg \mathrm{false}]\ (y \neq \mathrm{NULL}); \\ \mathrm{NULL}. \end{cases} \quad (16)$$

It should be noted that the semantics, presented here, does not take into consideration the possibility of application of meta-rules, providing a change of the rule choice strategy in the inference process, as well as a dynamic change in the set of rules themselves. Calculation of a set of parent concepts was carried out prior to the inference process of the first parent concept, which does not take into consideration the possibility of a change in this set in the process of subsequent inference.

Calculation of the value of expression *this.s* was performed as follows:

$$\|this.s\|^{\tau}_{\Phi \to \Phi'} = \|\tau.s\|^{\tau}_{\Phi \to \Phi'}. \quad (17)$$

And for the expression with the zero context:

$$\|\tau.s\|^{NULL}_{\Phi \to \Phi'} = \|\tau.s\|^{\tau}_{\Phi \to \Phi'}. \quad (18)$$

For description of the possibility of dynamic obtaining the values from the outside in the course of bottom-up inference, it is possible to use two approaches: the introduction of a set of pairs "question-answer" in the calculation context [42], or by the function-oracle [43]. In the latter case, we determine an external function of interaction with environment $A: \mathrm{D}^{\mathrm{set}} \to \mathrm{X}$, determined on a set of descriptors $\mathrm{D}^{\mathrm{set}}$, and the corresponding class of expressions, for which operation $\|.\|$ is introduced in the following way:

$$\|\mathrm{A(d)}\|_{\Phi \to \Phi} = \mathrm{A(d)}. \quad (19)$$

The use of this approach more naturally expresses the character of a dialogue, controlled by the inference process, that is, the calls of the function-oracle occur as often as required in the order, in which a user is asked questions. However, difficulties arise when studying semantics, making it non-deterministic, that is, the result depends on the behavior of external functions, not formalized in the model. The introduction of a set of answers of a user in the inferential context leads to deterministic semantics that does not reflect the dynamics of the questions asked (in this case, it is necessary to fix a set of "answers" beforehand). Note that a variant of this approach is complete rejection of the question

asking operation and replacement it with primary filling all known fields with values.

There are several approaches to formalization of the top-down inference. In the classic algorithm of the direct inference, a conflict set of the applied rules is formed at each step, of which one is applied, which leads to a change in the state. In the case of monotonous inference, the function of one rule application will be monotonous, which leads to a very simple description of the semantics with the use of a fixed point based on the Tarsky theorem. A similar approach in relation to logical programming is described in [43]. However, such a "classic" approach implies the use of the whole set of rules at each inference step and does not take into consideration the attachment to concept fields.

The research used the approach, in which the rules of top-down inference will be attached to concept fields. When a value is assigned to a field, there will operate only the rules, associated with this field, that is, in the left part of which the value in this field is found. Application of the rules would be consistent with the change in the state according to some function.

Each attached rule-daemon has the form $D=\langle E,A\rangle$, where $E\in E^{set}$ is the conditional part of the rule, which is an arbitrary expression, $A:\Phi^{set}\to\Phi^{set}$ is a function of a change in the state, which will be the function of a change in field value or a composition of such functions. It is necessary to parametrize these functions with current calculation context $C\in C^{set}$, then $A:C^{set}\to\Phi^{set}\to\Phi^{set}$.

It makes sense to consider the compositions of the following elementary functions:

$$[this.s\leftarrow v](C,\Phi)=\Phi[C.s\leftarrow v], \tag{20}$$

$$[\tau.s\leftarrow v](C,\Phi)=\Phi[\tau.s\leftarrow v]. \tag{21}$$

Let us introduce into consideration a set of modified slots $\Delta(A)$. for each function A. Obviously,

$$\Delta([\tau,s\leftarrow v])=\{\tau.s\}$$

and

$$\Delta(A_1,...,A_n)=\Delta(\{A_1,...,A_n\})=\bigcup_{i=1}^{n}\Delta(A_i).$$

To describe the top-down inference process, we will need the concept of activation function

$$\alpha:I^{set}\to\{free,todo,applied\},$$

where

$$\alpha(\tau.s)=\begin{cases}\text{free},\\ \text{todo},\\ \text{applied}.\end{cases} \tag{22}$$

A set of activation functions is A^{set}.

The order relationship was assigned on the set {free, todo, applied}:

$$\text{free}\subseteq\text{todo}\subseteq\text{applied}. \tag{23}$$

Then

$$\alpha_1\subseteq\alpha_2\Leftrightarrow\forall\langle\tau,s\rangle\in I\ \ \alpha_1(\tau,s)\subseteq\alpha_2(\tau,s). \tag{24}$$

For formalization of top-down inference, the function was used

$$F^c:A^{set}\times\Phi^{set}\to A^{set}\times\Phi^{set},$$

which by the function of activation and the state applies rules-daemons for this state. In this case, it marks such fields in the activation function that were changed during processing, as well as the fields with the same name for parents of this concept. Let us designate through

$$\alpha[\tau,s]=\lambda\langle\tau_x,s_x\rangle.(\langle\tau,s\rangle=\langle\tau_x,s_x\rangle)\to \\ \to(\text{todo}\vee\alpha(\tau,s));\ \alpha(\tau_x,s_x)). \tag{25}$$

It is possible to determine inductively

$$\alpha[\{\tau_i.s_i\}_{i=1}^{n}]=\ \alpha[\tau_1.s_1].....[\tau_n.s_n].$$

For application of set of rules $S\subset D^{set}$ to the state, we used function

$$\varphi^c_{\subseteq}:D^{set}\times A^{set}\times\Phi^{set}\to D^{set}\times A^{set}\times\Phi^{set}:$$

$$\varphi^c(S,\alpha,\Phi)=\begin{cases}\langle\varnothing,\alpha,\Phi\rangle,\ \text{if } S=\varnothing;\\ \langle S^-,\alpha[\inf S],\text{isTrue}\|E\|'_{\Phi\to\Phi}\to A(C)(\Phi);(\Phi)\rangle.\end{cases} \tag{26}$$

Conditions for ordering the initial and final states follow from monotony φ. The function $Anc_{\Phi\to\Phi'}(\tau)$, returning a set of parent concepts for τ, was described (in this case dynamic calculation of a parent requires foreseeing the possibility of a change in state Φ in the calculation process):

$$Anc_{\Phi\to\Phi'}(\tau)=\begin{cases}\varnothing,\ \text{if } \|\tau.parent\|_{\Phi\to\Phi'}=NULL;\\ \|\tau.parent\|_{\Phi\to\Phi_1}\cup\left(\bigcup\limits_{x\in\|\tau.parent\|_{\Phi\to\Phi''}}Anc_{\Phi i\to\Phi i+1}(x)\right);\\ \text{where } \Phi'=\Phi_N.\end{cases} \tag{27}$$

Considering the introduced designations, we will obtain

$$\Phi^c(\alpha,\Phi)=\begin{cases}\langle\alpha,\Phi\rangle,\ \text{if } \forall\langle\tau,s\rangle\in I^{set}\ \alpha(\tau,s)\neq\text{todo};\\ \langle\alpha',\Phi'\rangle,\ \text{where } \langle\alpha'',\Phi''\rangle=\phi^C_{\Phi(f.s)\subseteq d}(\Phi(\tau,s).\text{daemons},\alpha,\Phi),\\ \langle\tau,s\rangle=\min\{\langle\tau_x,s_x\rangle|\alpha(\tau_x,s_x)=\text{todo}\};\\ \alpha'''=\alpha''\left[\Delta(\Phi(\tau,s).\text{daemons})\cup\left(\bigcup\limits_{p\in Anc_{\Phi''\to\Phi'}(\tau)}\{p.s\}\right)\right];\\ \alpha'=\alpha'''[\tau.s\leftarrow\text{applied}].\end{cases} \tag{28}$$

Function Φ^c finds the first (in a sense of lexicographic ordering) field τ.s, marked in α as *todo* and uses φ for application of all rules-daemons related to it with the correspondent order relationship. Then it marks all the changed fields, as well as the fields with the same name of all parents *t* as candidates for subsequent processing. In conclusion it establishes value α(τ, s) as applied, which excludes re-use of daemons for this field.

In the proposed model, set of constraints $\{C_i\}_{i=1}^{N}$, each of which is an expression that is true for this field, is associated with each field. In these expressions, a reference to the current concept *this*, allowing organizing inheritance

of constraints for a field by a recursive call of parent field constraints, is used. Let us describe such semantics of constraints, in which the current state Φ does not change.

To check constraints, will introduce the check function:

$$C^{set} \times \Phi \times I^{set} \to \{true, false\},$$

determined as

$$\text{check}^{C}_{\tau.s}(\Phi) = isTrue(\|C_1\|^{C}_{\Phi\to\Phi} or .... or \|C_N\|^{C}_{\Phi\to\Phi},$$

where

$$\Phi(\tau, s).constra\,\text{int}\, s = \{C_i\}_{i=1}^{N}.$$

To check constraints taking into account inheritance, just as it was done above, we will introduce function $Anc'_{\Phi}(\tau)$, returning the set of parent concepts for τ, the values of which have already been obtained previously or assigned explicitly:

$$\text{Anc}_{\Phi}(\tau) = \begin{cases} \varnothing, \text{ if } \|\tau.\text{parent}\|'_{\Phi\to\Phi} = \text{NULL}, \\ \|\tau.\text{parent}\|'_{\Phi\to\Phi} \cup \left(\bigcup\limits_{x\in\|\tau.\text{parent}\|_{\Phi\to\Phi}} \text{Anc}'_{\Phi}(x) \right). \end{cases} \quad (29)$$

Then the function of complete constraint check is

$$\text{ensure}_{\tau.s}(\Phi) = \text{check}^{\tau}_{\tau.s}(\Phi) or \left(or_{p\in Anc_{\Phi}'(\tau)} \text{check}^{\tau}_{p.s}(\Phi) \right). \quad (30)$$

To determine constraint check semantics in the inference process, we will predetermine the function of a slot value change as follows:

$$\text{write}(\Phi, \langle \tau, s \rangle, v) \text{isTrue ensure}_{\tau.s}(\Phi[\tau.s \leftarrow v]) \to \Phi[\tau.s \leftarrow v]; \Phi. \quad (31)$$

By a combined inference we imply such an inference engine, at which top-down and bottom-up inferences are alternately applied. This approach combines the merits of two basic inference engines, and in addition, is more universal because leaves the possibility to apply both bottom-up and top-down inferences in a pure form and their combination. In this case, bottom-up inference determines the main route of motion by the decision tree and is responsible for obtaining the necessary information from the user in the inference process. A top-down inference allows "extending" a set of the obtained data due to what can be obtained from the already existing set without additional information.

We developed the approach, in which after each step of bottom-up inference, i. e., as soon as the value of a certain field is derived, a directed exhaustive inference will be applied. To describe such semantics, expression (16) should be written as follows:

$$\|\tau.s\|_{\Phi\to\Phi'} = \begin{cases} \Phi(\tau.s).\text{value}, \ \Phi(\tau.s).\text{value} \neq \text{NULL} \ (\Phi' = \Phi); \\ \text{NULL}, \ \Phi(\tau.s).\text{busy} = \text{true}; \\ x = \mu_{\Phi[\tau.s.\text{busy}\leftarrow\text{true}]\to\Phi''}(\langle \Phi(\tau, s).\text{rules}, \ \Phi(\tau, s). \subseteq_q \rangle), \\ \Phi' = F_{\tau.s}(\Phi''[\tau.s \leftarrow x, \ \tau.s.\text{busy} \leftarrow \text{false}]) \ (x \neq \text{NULL}); \\ y = \mu_{\Phi'''\to\Phi''''}(\{p.s | \ p \in \|\tau.\text{child}\|_{\Phi''\to\Phi'''}\}), \\ \Phi' = F_{\tau.s}(\Phi''''[\tau.s \leftarrow y, \ \tau.s.\text{busy} \leftarrow \text{false}]) \ (y \neq \text{NULL}); \\ \text{NULL}, \text{otherwise}. \end{cases} \quad (32)$$

The difference of the considered approach is that the function of directed top-down inference $F_{\tau.s}$ is applied to the final state after bottom-up inference. This function ensures operation of all relevant rules of top-down inference for a newly updated field, as well as for all fields, the values of which changed during top-down inference. If in this definition we will assume that function 31 checks constraints, we will obtain a description of inference semantics with the possibility of rollback on not meeting constraints. This rollback activates other rules for determining the filed value, but inference monotony will not be disrupted.

6. Discussion of results of the conducted study

The central results of the research stage, presented in this article are:

– a description of internal relations between concepts, integrating a set of concepts and fields in the language of operational semantics;

– introduction of external relations that characterize the structural relations of concepts, including hierarchical relations of aggregation and synthesis.

The usefulness of the obtained results is determined by the possibility of the adequate representation of dynamics of functioning of the organizational structure of an aviation enterprise due to the combination of generalization and aggregation relations. The results of this research can be used in informatization of a wide class of complex objects of mixed nature (socio-technical, techno-socio-economic, etc.).

The main disadvantage of the obtained results of the research is relatively high costs of development of information structures of the ontological nature. At the same time, the apparent advantage is the possibility of repeated, that is, multiple use of ontological information structures in making decisions on organization business processes at aviation enterprises. Using the obtained results allows enhancing effectiveness of production solutions and their operative making.

The described results are the methodological basis for the development of software for organization of inference on knowledge directly in environment of ontologies, which are proposed to use as a part of the core of production DSS. Subsequently, it is planned to develop the instrumental environment for the creation and support of the operation of the ontological system on formation of decisions, related to the organization of business processes at aviation enterprises.

7. Conclusions

1. The deductive inference engine with the use of the ontological knowledge model in the part of the sequence of formation of the inference chain was developed, which made it possible to take into consideration child relations between the concepts of a subject domain and, thereby, ensure knowledge correctness.

2. The combined inference engine, which, unlike the existing ones, is based on alternating use of top-down and bottom-up inferences in the environment of the ontological knowledge model, was developed, which gave an opportunity to eliminate incompleteness and inconsistency of knowledge.

References

1. Melihov A. N., Bernshteyn L. S., Korovin S. Ya. Situacionnye sovetuyushchie sistemy s nechetkoy logikoy. Moscow: Nauka, 1990. 271 p.
2. Aliev R. A., Abdikeev N. M., Shahnazarov M. M. Proizvodstvennye sistemy s iskusstvennym intellektom. Moscow: Radio i svyaz', 1990. 262 p.
3. Aliev R. A., Mamedova G. A. Identifikaciya i optimal'noe upravlenie nechetkimi dinamicheskimi sistemami // Izv. AN. Seriya: Tekhnicheskaya kibernetika. 1993. Issue 6. P. 1–9.
4. Anisimov V. Yu., Borisov E. V. Metody dostovernosti realizacii nechetkih otnosheniy v prikladnyh sistemah iskusstvennogo intellekta // Izv. AN. Seriya: Tekhnicheskaya kibernetika. 1991. Issue 5. P. 24–89.
5. Gorban' A. N., Rossiev D. A. Neyronnye seti na personal'nom komp'yutere. Novosibirsk: Nauka, 1996. 276 p.
6. Aliev R. A., Cerkovniy A. E., Mamedova G. A. Upravlenie proizvodstvom pri nechetkoy iskhodnoy informacii. Moscow: Energoatomizdat, 1991. 201 p.
7. Parallel'niy processor nechetkogo vyvoda dlya situacionnyh ekspertnyh sistem / Bershteyn L. S., Kazupeev V. M., Korovin S. Ya., Melihov A. I. // Izv. AN. Seriya: Tekhnicheskaya kibernetika. 1990. Issue 5. P. 86–90
8. Logicheskiy podhod k iskusstvennomu intellektu: Ot modal'noy logiki k logike baz dannyh. Moscow: Mir, 1998. 494 p.
9. Enterprise Interoperability VII: Enterprise Interoperability in the Digitized and Networked Factory of the Future / K. Mertins, R. Jardim-Gonçalves, K. Popplewell, J. P. Mendonça (Eds.). Springer, 2016. 344 p. doi: 10.1007/978-3-319-30957-6
10. Ahrehatsiya danykh dlia formuvannia vyrobnychykh rishen na promyslovykh pidpryiemstvakh iz vykorystanniam ontolohichnykh system / Shostak I., Sobchak A., Firsova H., Kushnarenko O. // Traiektoriya nauky. 2016. Issue 3 (8).
11. Ada Ş., Ghaffarzadeh M. Decision making based on management information system and decision support system // International Journal of Economics, Commerce and Management. 2015. Vol. III, Issue 4. P. 1–14.
12. Kruglov V. V. Iskusstvennye neyronnye seti: Teoriya i praktika. Moscow: Goryachaya liniya-Telekom, 2002. 382 p.
13. Danova M. A., Shostak I. V. Ontologicheskiy podhod k kompleksnoy komp'yuterizacii processa prognozirovaniya nauchno-tekhnicheskogo razvitiya regiona // Suchasni informatsiyni tekhnolohiyi v ekonomitsi ta upravlinni pidpryiemstvamy, prohramamy ta proektamy: tez. dop. X Mizhnar. nauk.-prakt. konf. Alushta, 2012. P. 60–61.
14. Shostak Y. V., Danova M. A. Analiz innovatsiynoi diyalnosti rehioniv zasobamy ontolohichnoho inzhynirynhu // Informacionnye sistemy i tekhnologii: tez. dokl. 6-y mezhdunar. nauchn.-tekhn. konf. Kharkiv, Koblevo, 2017. P. 57–58.
15. Kudelina D. B, Shostak I. V., Gruzdo I. V. Upravlenie znaniyami razrabotchikov softvernoy firmy po sertifikacii programmnyh produktov na osnove ontologicheskogo podhoda // Systemy obrobky informatsiyi. 2016. Issue 5 (142). P. 50–55.
16. Modeli ontologiy i ontologicheskoy sistemy podderzhki prinyatiya resheniy po vyboru ruchnyh impul'snyh ustroystv / Vorob'ev Yu. A., Nechiporuk N. V., Kobrin V. N., Shostak I. V. // Naukovi notatky. 2014. Issue 46. P. 77–83.
17. Shostak I., Butenko I. Ontology approach to realization of information technology for normative profile formimg at critical software certification // Zbirnyk naukovykh prats viyskovoho instytutu KNU im. T.H. Shevchenko. 2012. Issue 38. P. 250–253.
18. Cvetkov V. Ya. Kognitivnoe upravlenie. Moscow: MAKS Press, 2017. 69 p.
19. Katalnikova S., Novickis L. Choice of Knowledge Representation Model for Development of Knowledge Base: Possible Solutions // International Journal of Advanced Computer Science and Applications. 2018. Vol. 9, Issue 2. doi: 10.14569/ijacsa.2018.090249
20. Gluhih I. N., Akhmadulin R. K. Problem-Oriented Corporate Knowledge Base Models on the Case-Based Reasoning Approach Basis // IOP Conference Series: Materials Science and Engineering. 2017. Vol. 221. P. 012025. doi: 10.1088/1755-1315/221/1/012025
21. Rashid P. Q. Semantic Network and Frame Knowledge Representation Formalisms in Artificial Intelligence. Gazimağusa, 2015. 60 p. URL: https://pdfs.semanticscholar.org/3050/f186dfd77fce3ab3d094abebd78411f5a0c1.pdf
22. Ramirez C., Valdes B. A General Knowledge Representation Model of Concepts // Advances in Knowledge Representation. 2012. URL: http://cdn.intechopen.com/pdfs/36656/InTech-A_general_knowledge_representation_model_of_concepts.pdf
23. An Ontological Approach for Domain Knowledge Modeling and Management in E-Learning Systems / Panagiotopoulos I., Kalou A., Pierrakeas C., Kameas A. // IFIP Advances in Information and Communication Technology. 2012. P. 95–104. doi: 10.1007/978-3-642-33412-2_10
24. Osipov G. S. Priobretenie znaniy intellektual'nymi sistemami. Moscow: Nauka, 1997. 345 p.
25. Lyuger D. F. Iskusstvennyy intellekt. Strategii i metody resheniya slozhnyh problem. Moscow: Vil'yams, 2005. 864 p.
26. Rassel S., Norvig P. Iskusstvenniy intellekt: sovremennyy podhod. Moscow: Vil'yams, 2017. 1408 p.
27. Allen J. E., Ferguson G. Actions and events in interval temporal logic // Technical Report 521. Rochester University, 1994. URL: https://urresearch.rochester.edu/fileDownloadForInstitutionalItem.action?itemId=609&itemFileId=736
28. Intelligent Control Systems / Vassilyev S. N., Kelina A. Y., Kudinov Y. I., Pashchenko F. F. // Procedia Computer Science. 2017. Vol. 103. P. 623–628. doi: 10.1016/j.procs.2017.01.088

29. Argente E., Julian V., Botti V. Multi-Agent System Development Based on Organizations // Electronic Notes in Theoretical Computer Science. 2006. Vol. 150, Issue 3. P. 55–71. doi: 10.1016/j.entcs.2006.03.005
30. Yakovlev M. A. Ekspertnye sistemy s primeneniem dialogovogo interfeysa na estestvennom yazyke // Uchenye zametki TOGU. 2013. Vol. 4, Issue 3. P. 31–39.
31. Tarasov V. B. Logiko-lingvisticheskie modeli v iskusstvennom intellekte: proshloe, nastoyashchee, budushchee. URL: http://textanalysis.ru/jce/details/instrument/doc_view/186-logiko-lingvisticheskie-modeli
32. Gavrilova T. A., Horoshevskiy V. F. Bazy znaniy intellektual'nyh sistem. Sankt-Peterburg: Piter, 2001. 170 p.
33. Dyubua D., Prad A. Teoriya vozmozhnostey. Prilozheniya k predstavleniyu znaniy v informatike. Moscow: Radio i svyaz', 1990. 288 p.
34. Jones M. N., Willits J., Dennis S. Models of Semantic Memory. 2015. URL: http://www.languagelearninglab.org/uploads/5/3/5/7/53575061/jones_willits_dennis_2015.pdf
35. Grekhem I. Ob'ektno-orientirovannye metody: principy i praktika. 3-e izd. Moscow: Vil'yams, 2004. 879 p.
36. Schank R. C., Robert P. A. Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. New Jersey: Lawrence Erlbaum Associates, 1977. 256 p. doi: 10.4324/9780203781036
37. Gaeta M., Orciuoli F., Ritrovato P. Advanced ontology management system for personalised e-Learning // Knowledge-Based Systems. 2009. Vol. 22, Issue 4. P. 292–301. doi: 10.1016/j.knosys.2009.01.006
38. Knowledge-Based Interactive Postmining of Association Rules Using Ontologies // IEEE Transactions on Knowledge and Data Engineering. 2010. Vol. 22, Issue 6. P. 784–797. doi: 10.1109/tkde.2010.29
39. Karapiperis S., Apostolou D. Consensus building in collaborative ontology engineering process // Journal of Universal Knowledge Management. 2006. Vol. 1. P. 199–216.
40. Lapshin V. Ontologiya v komp'yuternyh sistemah. Moscow: Nauchniy mir, 2010. 224 p.
41. Avtomaticheskaya obrabotka tekstov na estestvennom yazyke i analiz dannyh: ucheb. pos. / Bol'shakova E. I., Voroncov K. V., Efremova N. E., Klyshinskiy E. S., Lukashevich N. V., Sapin A. S. Moscow: Izd-vo NIU VSHE, 2017. 269 p.
42. Marcenyuk M. A. Matrichnoe predstavlenie nechetkoy logiki // Nechetkie sistemy i myagkie vychisleniya. 2007. Vol. 2, Issue 3. P. 7–36.
43. Nguen M. H. Modelirovanie s pomoshch'yu nechetko-znachnoy veroyatnostnoy logiki // Izv. AN. Seriya: Tekhnicheskaya kibernetika. 1993. Issue 5. P. 128–143.
44. Vittih V. A. Upravlenie otkrytymi sistemami na osnove integracii znaniy // Avtometriya. 1999. Issue 3. P. 38–49.
45. Kotis K., Vouros G. A. Human-centered ontology engineering: The HCOME methodology // Knowledge and Information Systems. 2005. Vol. 10, Issue 1. P. 109–131. doi: 10.1007/s10115-005-0227-4