Model of the ontological representation of knowledge about the information security incidents[*]

Denis N. Biryukov[1[0000-0003-1300-2470]], Alexander P. Glukhov[2[0000-0001-5368-4109]], Anatoly A. Kornienko [2[0000-0002-6076-7241]]

[1]Mozhaisky Military Aerospace Academy, Zhdanovskaya street 13, Saint Petersburg, Russia, 197198
Biryukov.d.n@ya.ru
[2]Emperor Alexander I St. Petersburg State Transport University, Moskovsky prospekt 9, Saint Petersburg, Russia, 190031

Abstract. The approach to the ontological description of any subject domain based on application of concepts of three types is offered: "Objects", "Properties" and "Actions". Thus various aspects of representations used for the description of knowledge are offered to be ordered partially property of approximation in functional layers, segments and areas. It has to allow to model semantic features of context-dependent knowledge of subject domains, to consider their changes and specifications at generation of decisions. That will open possibilities of forecasting of intentions and prevention of realization of cyberthreats to critical information infrastructure.

Keywords: knowledge, context, ontology, approximation, memory denotational semantics, conflict, cybersistem.

1 Introduction

Information and communication technologies are being used to strengthen the human intellect, complementing its creativity in working with information. Researchers predict that as this area develops, the "external cortex" of the brain ("exocortex") will be formed, which is the system of programs that complement and expand the human thought processes [1].

Obviously, the more intelligent the desired programs will be, the more tasks in searching for solutions can be transferred to them by a person, especially in the field of information security while protecting critical information infrastructure from attacking influences. It can also be assumed that the intellectuality of the designed and developed programs essentially depends on their ability to carry out a semantic, context-related, knowledge and data processing. The priority is the development of systems that are capable of semantic processing of incoming and accumulated data and knowledge and synthesize at first possible and then more and more accurate solutions that are context-

[*] Copyright 2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

dependent on conditions of tasks. The desired system should be able to synthesize scenarios of pre-emptive behavior in the conflict, which should contribute to predicting intentions and preventing cyber-attacks on elements of the critical information infrastructure.

2 The approach to the modeling of context-dependent ontologies

Taking into account the peculiarities of data organization in human semantic memory [2-5], it is advisable to represent knowledge of the problem area (PA) in the form of a hierarchy of structured objects related to each other by relationships. This idea is based on such models of representations as frames, semantic networks, UML, etc. Unfortunately, although all these languages are quite a convenient means for presenting knowledge about the PA of the conflict, they do not have the ability to reflect semantics, and the information expressed in them is intended more for human rather than machine perception. Also the logic of predicates (FOL) gained a great development, which contains mechanisms of semantic processing, but has no convenient means of representing knowledge.

In 1985, R. Brahman offered to combine semantic networks and FOL. The result was called as terminological logic, and then, in the process of development, it turned into a family of descriptive logics [6], which found their practical application in describing ontologies in the currently developing semantic Web.

The ontology describes the basic concepts (conditions) of the subject area and defines the relationships between them. For the description of ontologies, appropriate languages are used, for example - OWL (Web Ontology Language).

At the initial stage, during the construction of the ontology, it is necessary to specify a list of Concepts (unary predicates) and Roles (binary predicates), which would allow us to describe the PA of conflict in the future taking into account the possible contexts. When setting the specific roles used for ontological modeling, the description of each role can be refined by specifying certain characteristics. Depending on what characteristics have been entered (used), there can be used one or another (from the positions of the selected logic) machine.

It should be separately noted that the ontological approach to the description of random PA does not introduce certain restrictions on the order of selection and use of certain types of Concepts and Roles (except for the requirements for the correct use of roles). In practice this leads to the fact that knowledge engineers use a large number of different roles for describing different PAs, which, in the best case, does not contribute, and more often makes it impossible to combine different ontologies (ontologies describing different PAs and created by various specialists). In the same cases in which such an association is possible, it is often needed to change the logical output engine used to generate new knowledge over a new (unified) ontology. The possibility of integrating ontologies describing different PA of the conflict is simply necessary for the projected intellectual system (IS) to have the potential ability to learn, which means transferring knowledge about conflicts and their resolution from one PA to another (for example: from "bio- sphere "in the" cyber-sphere ").

In order for the projected system to synthesize scenarios of pre-emptive behavior in conflicts, it must be able to represent and process knowledge about the causes of conflicts, their processes and the consequences of conflicts. For this it's necessary for it to have the ability to represent knowledge about objects, their properties and the processes of interaction of different objects and subjects. At the same time, it should be taken into account that the problem areas in should not introduce restrictions that lead to the impossibility of describing knowledge from these PA in a single ontology. Since otherwise, this can lead to the fundamental impossibility of enriching the information system with knowledge of behavior in conflicts that occur in other PAs.

An important issue in the manipulation of knowledge represented through a single ontology in the Knowledge Base (KB) of IS is the issue associated with the allocation of those fragments from ontology, the knowledge of which should be available to the system at any time in solving particular problems (in different contexts).

In view of this, it is possible to formulate a number of requirements that must be taken into account when creating a meta-modeling system, representing and manipulating knowledge. The knowledge meta-modeling system must be capable of:

representing and processing knowledge from various problem areas about objects, their properties and processes (for the possibility of enriching the KB with descriptions of various conflicts that can occur in different PAs);

containing a limited number of types of Concepts and Roles, sufficient for describing arbitrary PAs (for setting unified rules for building an ontology that further consolidates them, and applying uniform logical rules for generating new knowledge regardless of the specifics of the PA and the conflicts under consideration);

having solvable algorithms for directional searching of data (to realize the possibility of determining the necessary fragment of an ontology based on tasks and contexts);

searching for similarity of fragments of ontology by analogy (for search by similarity analogy of emerging tasks and their solutions in the course of conflicts in different PAs);

changing the availability of knowledge for further processing (to sort the obtained solutions, to identify the most associated knowledge, and to "forget" false and little used / little-proven knowledge).

The needed abilities require a formal definition of knowledge contexts' denotational semantics in the ontological modeling of conflict's problem areas.

The issue related to endowing the programming languages with mathematical semantics was raised in 1971 by D. Scott [11], but to this day, the mathematical theory he offered has not been developed to a level that allows him to apply it to solve specific practical tasks at the proper level.

As the central element of the mathematical semantics introduced by D. Scott, you can point to abstract functions of the elements of the data type that is associated with the input variables, and take the values of the elements of the data type that is associated with the output. Obviously, when switching to a more "abstract" description of the tasks being solved at the level of functions and input / output data at the type level, it is impossible to completely evade the operational aspects. That's because eventually the functions must be implemented and executed on computational means capable of manipulating only finite configurations. At the same time, mathematical semantics should allow for the operational modeling of abstract entities.

In order for manipulation of abstractions to become possible, it is necessary to have an appropriate mathematically defined language. In [11] D. Scott offered a noteworthy view on the nature of data types and functions (mappings) from one data type to another. He also pointed out that in mathematics the questions connected with the description of functions defined on all admissible functions as arguments and applicable even to themselves as to the argument were not sufficiently worked out and offered a mathematical theory of functions that can be used as a project giving "correct" approach to semantics.

Further, in [12], D. Scott, using grids described the abstract theory of finite approximation and infinite limits in general terms. He used these concepts to effectively build a class of spaces (data types), including functional ones, which allowed them to be used as a space of mathematical meanings in semantic interpretations of high-level programming languages.

In the article [13], A.Shamir and W. Wage, relying on the formalisms introduced by D. Scott, presented a new approach to the semantics of data types, in which types themselves are included as elements in the area of objects. This approach allows types to have subtypes, allows to examine truly polymorphic functions and gives exact semantics for recursive type definitions. In addition, this approach provides simple and direct methods for proving the typical properties of recursive definitions.

Some scientists [14-18] advocated for an algebraic analysis of data types, which emphasized the fact that the data type consists not only of a set (sometimes partially ordered), but also of operations that satisfy certain equalities.

A. Demers, J. Donahue, R. Teitelbaum and J. Williams also pointed out [19] that data types must be considered inextricably with the operations implemented on them.

Thus, we should point out the existence of two points of view on the functions themselves, namely, the extensional and intensional points of view. With an extensional view of functions, they are treated as abstract objects, which are completely determined by their pairs (argument, value), i.e. their schedules. It implies the axiom of extensionality feasibility, that says if $f(x) = g(x)$ for all x, than $f = g$.

This point of view is opposed by the intensional viewpoint, which is very significant [20] in the context of computability and programming, according to which the function is an "operation", the rule of "transformation" or "calculation," some definition of such rule.

It is also necessary to dwell in detail on Ch.Hoar's theory of types [21], which is based on the concept of a type whose distinctive features are the following:

1) type defines the class of values that a variable or expression can take;

2) each value belongs to one and only one type;

3) the type of the constant's, variable's or expression's value can be derived either from the context or from the form of the operand itself, without reference to the values computed during program execution;

4) each operation corresponds to some fixed type of its operands and some fixed type of result;

5) for each type the properties of values and elementary operations on values are given by means of axioms;

6) when working with a high-level language, knowledge of the type makes it possible to detect meaningless constructions in the program and to solve the problem of the method of representing data and converting it in a computer;

7) the types we are interested in are types familiar to mathematicians: cartesian product, disjoint unions, sets, functions, sequences, and recursive structures.

A number of propositions indicated by Ch.Hoar are controversial (for example, position 2, since a number of authors believe that some types can be subsets of other types [22-24] or that an object can belong to a finite number of types. [25] This question is discussed in [13, 26, 27] and is part of a more general question of the relationship between types), but the proposition that states that for each type of property values and elementary operations on values are given by axioms - deserves special attention.

3 Offers for the presentation of knowledge for the intellectual system

In [11] D. Scott came to the conclusion that the function space should also be considered as having a data type. Since the function is, in general, an infinite object itself, D. Scott has offered the idea of finite approximation. He claimed [28] that there is a general theory of finite approximation and there are many types of objects that can be obtained as limits of approximation.

The mathematical theory proposed by D. Scott is based on the idea that the approximation relation ($\sqsubseteq$) should be a partial order, and not any structure with a partial order is suitable, but one in which limits can be taken.

Note $x \sqsubseteq y$ ¬ means that x approximates y. This is a qualitative relationship from which it must follow that x is compatible with y. It can be argued that x is worse, and y is better (or vice versa), but we cannot say how close x is to y.

In subsequent discussions, it is assumed that types (at least) are structured by relationships. In accordance with the intuitive understanding, it should be considered that the ratio ϕ is:

$$\text{reflexive } \forall a(a \ \phi \ a) \tag{1}$$

$$\text{transitive } (\forall a, b, c \ (a \ \phi \ b) \land (b \ \phi \ c) \Longrightarrow a \ \phi \ c) \tag{2}$$

and

$$\text{antisymmetric } \forall a, b \ \ (a \ \phi \ b) \land (b \ \phi \ \text{a}) \Longrightarrow a = \text{b} \tag{3}$$

D. Scott introduced an axiom that states that: "The data type is a partially ordered set." In fact, it can be assumed that the entire set of elements is bounded from above by an element ⊤, which means $x \sqsubseteq \top$ always takes place. You can also enter an object ⊥ , for which it is always true that $\bot \sqsubseteq x$. In addition, we can assume that any two elements x and y have the *least upper bound* – l.u.b. ($x \sqcup y$) and the *greatest lower bound* – g.l.b.

($x \sqcap y$). An example of graphical representation of a partially ordered set of data types is shown in Fig. 1.

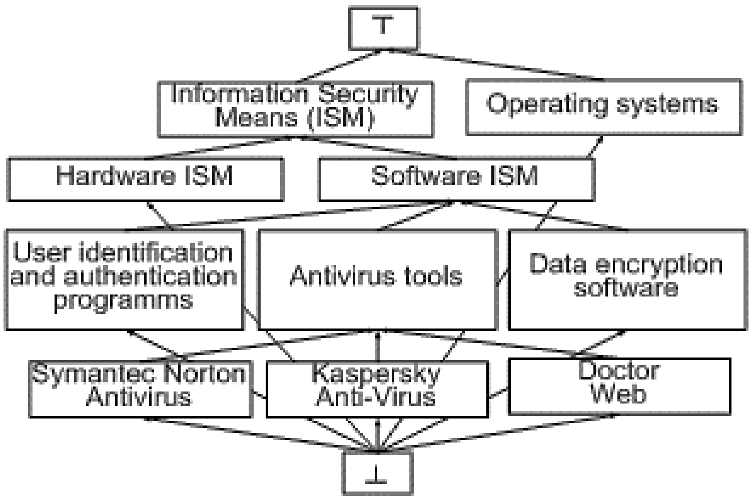

Fig. 1. An example of graphical representation of a partially ordered set of data types.

After it was assumed that the data type is within limits, it is necessary to revise the look at the functions. If the function is computable in some intuitive sense, then to extract the "finite" volume of information from any value of the function, it is necessary to ask for only a "finite" amount of information about the arguments (herein this case the concept of information is more qualitative rather than quantitative). However, it is still possible to express this fundamental restriction on functions: namely, functions must preserve limits. The mappings between the data types are continuous.

In the generality this can be expressed exactly in terms of directed sets. Subset $X \subseteq D$ is directed if every subset X has at least one upper bound in X. It should be noted that the directed set is not empty. Function $f: D \rightarrow D'$ is continuous in that and only that case if for any directed set $X \subseteq D$:

$$f(\sqcup \mathrm{X}) = \sqcup \{f(\mathrm{x}): \mathrm{x} \in \mathrm{X}\} \tag{4}$$

Not all l.u.b. should be considered as limits, but only l.u.b. of directed sets. It should be noted that the concept of continuity is easily extended to functions of several variables. For the continuity of a function with respect to all variables, it is simultaneously sufficient to require its continuity with respect to each variable separately [12].

Thus, in more abstract terms for any two complete lattices D and D′ it is possible to form a functional space $[D \rightarrow D']$ of all continuous functions from the first to the second.

Further, D. Scott [28] suggested that the partial order is a lattice, and the data type is a complete lattice in its partial order (this is another important axiom introduced by D. Scott.

The type of data satisfying the two above axioms introduced by D. Scott can be regarded as a topological space [28], and in such spaces the open set contains everything that approximates it along with each element.

This conclusion was developed in [13], in which the authors offered to combine all objects and data types together into a unified area. Any element of the region obtained like this appears then in two roles:

1)it is an object on which functions can be defined, including functions that are the smallest fixed points of recursive definitions;

2) it is the type of all objects that approximate it; therefore, the statement "x is of type y" and "any object of type x is an object of type y" is equivalent.

From an arbitrary initial region of "O" objects and a set of intuitive data types, you can form a new region “$\hat{O}$” (which is called a typical extension of the "O" region), adding types as new objects in the sense of (1). The extended relation $\sqsubseteq$ to "$\hat{O}$” is determined by the equivalence from (2): the object type is placed above objects of this type and over its subtypes (so $\sqsubseteq$ that it simultaneously sorts the types by inclusion and objects by approximation).

An important concept in describing any PA is the concept of the field itself. A region is meant to be [13] a partially ordered set D such that D has the smallest element and any directed set of elements of D has l.u.b. (such regions are usually called complete partial orders).

By the data type over D we mean a subset x of the universum D such that x is:

(1) closed down, which means if d_0 and d_1 belong to D and if $d_0 \sqsubseteq d_1$ than $d_1 \in x$ entails $d_0 \in x$;

(2) is closed with the respect to taking the upper bounds, which means if s is directed subset of x, than $\bigsqcup s \in x$.

In the modeling of semantic computations, the use of functional types seems very promising. D. Scott also considered [29] as very important that part of his research, in which he studied functions whose arguments are also functions, i.e. functions of higher orders. It was D. Scott who was the one of the first to use the functional space as a semantic structure.

To simplify the perception and use of functional types in practice, it is useful to be able to explicitly specify their partially ordered set in the form of Concepts of the appropriate type, as well as the relationships between them. This should facilitate the process of ontological modeling of conflict’s problem areas.

Based on the results of the analysis, it can be assumed that any type of data used for ontological modeling of problem areas of conflict can be described through a list of (1) objects that correspond to it; (2) properties of these objects and functions (and for the general case - (3) actions) that the objects under consideration can fulfill [30].

If we consider abstract classes of objects "O", properties "P" and actions "A", then in aggregate they can be represented in the form of a lattice containing one lower, one upper element and three discretely separated elements located on one same "horizontal".

It should be noted that if we take into account the fact that an object can consist of objects of a lower hierarchy (simpler objects) and be part of objects of a higher hierarchy (structurally more complex objects), then in general it is possible to speak about structures, rather than objects. However, for the sake of simplicity of the presentation, it is further proposed to use the term "objects", thus denoting some sort of "survey slice" in the context of certain contexts.

Thus, may: "O" – be lattice of "Objects", "P" – be lattice of "Properties", "A" – be lattice of "Actions".

Based on the analysis of the order of human intellectual activity, it can be assumed that the search for solutions is carried out in the plane of the "Properties" of the objects.

When constructing the lattice "Properties" it is proposed to use the relation "Approximates" ($F1_1$: $[P \to P]$ – «Property» Approximates « Property »).

In order to be able to hierarchically represent data about objects and their possible actions, it is suggested to enter mappings $F1_2$ ($F1_2$: $[O \to O]$ – «Object» Approximates «Object») and $F1_3$ ($F1_3$: $[A \to A]$ – «Action» Approximates «Action»)

Obviously, objects are judged through their properties, which should be considered as attributes of objects. Therefore, we can define a mapping F2 (F2: $[O \to P]$ – «Object» has «Property»).

It should be noted that the division of "Objects" into subclasses can be done in various ways (depending on which property is selected for the classification criterion). The possibility of different classification of properties (and therefore objects and actions) generates the possibility of stratification of knowledge.

Implementation of solutions is carried out through the implementation of "Actions", the point of which can be reduced to the modification of objects (or to the measurement and / or evaluation of their properties).

In order for the interaction between the two objects to become possible, they must have the appropriate properties. In other words, the presence of a certain property of an object generates the ability to perform a certain action (a certain class of actions). The same action performed by the first object can be suitable for influencing a certain property of the second object. Thus, we can define two more functions-arrows on lattices "Properties" and "Actions": F3 (F3: $[P \to A]$ – «Property» generates the ability for «Action») and F4 (F4: $[P \to A]$ – «Action» is suitable for impact on «Property»).

Data sets of functional types (F3 и F4) also make lattices.

The interaction of two objects (O_1 and O_2) can be formally described as follows:

$$[[O_1 \to P_1] \to A_1] \to [O_2 \to P_2] \tag{5}$$

Which means: "Object O_1, which has the property P_1, makes impact A_1 on object O_2, because the last one has the property P_2".

The above construction, which makes it possible to construct functional spaces, is very important, since its analysis (both by the knowledge engineer and the IS itself) allows answering the questions:

- "who (what) interacts" – (O_1 and O_2),
- "how does it interact" – (A_1),
- "why the interaction is possible" – ($O_1 \to P_1$ and $O_2 \to P_2$).

As O_1 , O_2, P_1, P_2 and A_1 are the elements of the corresponding partially ordered sets ("O", "P" и "A"), then this gives the prerequisites for the possibility of a systematic consideration of the described process of "interaction" and, subsequently, inductive and deductive deduce of "similar" processes.

In the overwhelming majority of cases, in practice, when constructing ontologies, they use constructions of the type: $O_1 \overset{A_1}{\rightarrow} O_2$, where O_1 and O_2 – Concepts describing interacting objects, and A_1 – A role with a semantically colored name. The use of such constructions is not always acceptable, since in some cases the semantics of Role is lost for the intellectual system that processes knowledge. If the Role is represented by a construction as: $P_1 \rightarrow A_1 \rightarrow P_2$, then this construction allows to take into account the additional context extracted from the location P_1, P_2 and A_1 in corresponding lattices.

In the 1960s, D. McCarthy [31] offered the calculation of situations, the purpose of which was to find a way to describe the results of operations (operations), regardless of the problem area. The basic formalism of situational calculus is an expression of the form s`=result (e,s) , where s` - situation that occurs when an event e occurs in situation s. In other words, the question related to the need to formalize the notion of context was identified. Obviously, the projected IS is simply obliged to make certain decisions taking into account the context, since it is the context that influences the evaluation of the values of the properties measured by the cyber system. Then the interaction of two objects in the certain context can be formally represented in a general form as in Fig.2, where $[O_3 \rightarrow P_3]$ – a generalized description of the context (which, if necessary, can be "deployed" into an arbitrarily complex interaction of objects surrounding the system of reality), A_{31} and A_{32} – the impact of the surrounding reality (context) on the first and second objects, respectively.

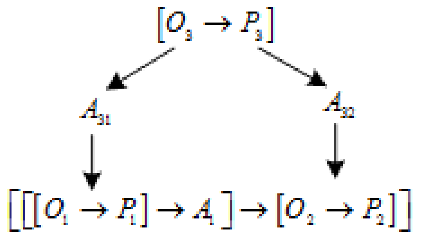
$$\begin{tikzcd}
 & {[O_3 \rightarrow P_3]} \arrow[dl] \arrow[dr] & \\
A_{31} \arrow[d] & & A_{32} \arrow[d] \\
{[[[O_1 \rightarrow P_1] \rightarrow A_1]} & & {\rightarrow [O_2 \rightarrow P_2]]}
\end{tikzcd}$$

Fig. 2. Representation of the interaction of two objects in a certain context

In order to make it possible to describe (output) processes, you must enter a mapping on the action lattice: F5 (F5: $[A \rightarrow A]$ – «Action» follows «Action»).

Undoubtedly, an important element in cognition is decomposition, so it is suggested to introduce a mapping on the lattice of objects: F6 (F6: $[O \rightarrow O]$– «Object» consists of «Object»).

It is also useful to introduce a similar mapping over the lattices of "Actions", which, if necessary, will allow the decomposition of the process under consideration into its component actions: F7 (F7: $[A \rightarrow A]$– «Action» consists of «Action»).

To be able to reflect the process of changing objects during their interaction, it is suggested to enter a mapping F8 (F8: $[O \rightarrow O]$ – «Object» produces «Object»).

It should be noted that the offered order of the structural organization of data in the memory of the Gyromat corresponds to the order of organization and processing of data by a person, since it is proved that he separately stores data (1) about events and sequences of events (scenarios) and (2) about objects and their properties (skeletons). It is also proved [4] that in semantic memory, information about objects is organized on the basis of differences between sensory or visual images and functional properties, which fully corresponds to the offered approach to construction of lattices "Objects" and "Actions" through the construction of the lattice "Properties".

4 Conclusion

In order for the projected cyber system to become truly intellectual, it must be able to structurally store data, knowledge and its contextual binding to tasks and solutions, and also perform complex cognitive actions ("logically think") with the aim of generating new knowledge not laid down in her directly, but acquired by it in the processes of setting and solving possible problems. The offered approach to ontological representation of data in the memory of an intellectual system, based on the theory of types and the theory of finite approximation, opens the possibility of a formal presentation of context-dependent multilayered knowledge about problem areas potentially containing knowledge of the strategies of behavior of the conflicting parties. This approach should allow organizing in the future the correct procedure for realizing plausible reasoning over context-dependent knowledge presented in the memory of cyber-threat prevention systems [34].

Acknowledgments
The study was performed with financial support RFBR, research project No. 17-20-03048

References

1. V. Pride., D.A. Medvedev. (2014). NBIC convergence phenomenon: Reality and expectations. [Online]. Available: http://www.nanonewsnet.ru/articles/2010/fenomen-nbic-konvergentsii-realnost-ozhidaniya.
2. A. Martin., L.L. Chao, “Semantic memory and the brain: Structure and Processes. Current Opinion in Neurobiology”, vol. 11, 2001 pp. 194–201.
3. J.F. Marques., N. Canessa., S. Siri, E. Catricala, S. Cappa, “Conceptual knowledge in the brain: fMRI evidence for a featural organization”, Brain Research, vol. 1194, 2008, pp. 90–99.
4. E. Tulving., “Episodic and semantic memory. Organization of Memory”, New York: Academic Press, 1972, pp. 381–403.
5. A.M Collins., M.R Quillian, “Retrieval time from semantic memory”, Journal of Verbal Learning and Verbal Behavior. 1969. vol. 8. pp. 240–247.
6. (2014) About formal bases of OWL. [Online]. Available: http://semanticfuture.net/index.php.
7. (2017) Ontology. [Online]. Available: http://www.aiportal.ru/articles/other/ontology.html.
8. (2014) OWL, language of web ontologies. Short review. Recommendation of W3C. 2004. [Online]. Available: http://www.thalion.kiev.ua/idx.php/7/009/article/#s4

9. D.A. Norman., T. Shallice, “Attention to action: Willed and automatic control of behavior”, Consciousness and Self-regulation. Advances in Research and Theory. New York: Plenum Press, 1986, vol. 4. pp. 1–18.
10. A.D. Baddeley, “The episodic buffer: A new component of working memory?”, Trends in Cognitive Sciences, 2000, vol. 4(11), pp. 417–423.
11. D. S. Scott, “Models for various type-free calculi” Logic, Methodology and Philosophy of Science IV (Proc. Int. Congress 1971), North-Holland, 1973, pp. 157–188.
12. D.S. Scott, “Outline of mathematical theory”, 4th Annual Princeton Conf. on Information Sciences and Systems, Princeton University, 1970, pp. 169–176.
13. A. Shamir, “Data types as objects”, Springer Berlin Heidelberg, 1977, pp. 465–479.
14. R. Burstall, “Programs and their proofs: an algebraic approach”, Machine Intelligence. Edinburgh Univ. Press, 1969, no. 4.
15. R. Burstall, “The algebraic theory of recursive program schemes”, Category Theory Applied to Computation and Control, 1974, no. 25, pp. 126–131.
16. J. Goguen, “Abstract data types as initial algebras and the correctness of data representation”, Current Trends in Progr. Methodology, IV. Data Structuring. Prentice-Hall, 1978, pp. 80–144.
17. J. Guttag, “Abstract data types and software validation”, Communications of the ACM, 1978, vol. 21(12), pp. 1048–1064.
18. S. Zilles, “Algebraic specifications for data types”, IBM Research Laboratory, San Jose, California, 1975.
19. A.J. Demers, “Incapsulated Data Types and Generic Procedures”, Design and Implementation of Programming Languages, LNCS 1977, no. 54, pp. 174–214.
20. A. Church, “The calculi of lambda-conversion”, Annals of Math. Studies, 1951, no. 6.
21. C.A.R. Hoare, “Notes on data structuring”, Structured Programming. Academic Press, 1972, pp. 98–197.
22. D.L Parnas, “Abstract types definedas classes of variables”, Proceedings Conference on Data: Abstraction, Definition, and Structure, Salt Lake City, 1976, pp. 149–154.
23. M.A. Kaplan, “A general scheme for the automatic inference of variable types”, Proceedings of the 5th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, 1978. pp. 60–75.
24. S.S Lavrov, “Basic concepts and designs of programming languages” M.: Statistica, 1981, (In Russ).
25. H. F. Ledgard, “Ten mini-languages: a study of topical issues in programming languages”, Corp. Surveys, 1971, no. 3, pp. 115–147.
26. C.H. Lewis, “Recursively defined data types”, Proceedings of the 1st annual ACM SIGACT-SIGPLAN symposium on Principles of programming languages, 1973, pp. 125–138.
27. D.M. Berry, “Type equivalence in strongly typed languages: one more look”, SIGPLAN Notices, 1979, no. 9, pp. 35–41.
28. D.S. Scott, “Lattice Theory. Data Types and Semantics”, Formal Semantics of Programming Languages, Prentice-Hall, Englewood Cliffs, N.J, 1972.
29. D.S. Scott, “Logic and programming languages”, Lectures of winners of an award of Turing., M.: Mir. 1993, pp. 65-83. (In Russ).
30. D.N. Biryukov, A.G Lomako, “Formalization of semantics for representation of knowledge of behavior of conflicting parties”, Materials of the 22nd scientific and practical conference "Methods and Technical Means of Safety of Information", SPB: St. Petersburg Polytechnical University, 2013, pp. 8-11. (In Russ).
31. J. McCarthy, P.J. Hayes, “Some philosophical problems from the standpoint of artificial intelligence”, Machine Intelligence, American Elsevier, New York, 1969, no. 4.

32. D.N. Biryukov, A.G. Lomako, “Approach to creation of the IB-systems capable to synthesize scenarios of anticipatory behavior in the information conflict*”,* Data protection. INSIDE, 2014, no. 6, pp. 42-50. (In Russ).
33. V.A. Bocharov, V.I. Markin, “Logic bases”, M.:MSU, 2008.
34. D.N. Biryukov, A.G. Lomako, “Approach to creation of system of cyber-threats preventing”, Problems of information security, Computer systems, SPB: St. Petersburg Polytechnical University, 2013, no. 2, pp. 13-19. (In Russ).