Requirements on a computer bank of knowledge

Alexander S. Kleschev[a] and Vasiliy A. Orlov[b]

Expert System Department
Institute for Automation & Control Processes
Far Eastern Branch of the Russian Academy of Sciences
5 Radio Street, 690041, Vladivostok, Russia
Tel: +7-4232-314001, Fax: +7-4232-310452
E-mail: [a] kleschev@iacp.dvo.ru, [b] v_orlov@yahoo.com
Web site: www.iacp.dvo.ru/es/

Abstract

Different kinds of information are used when solving tasks that arise in the life cycle of an applied knowledge based system (KBS). Many of these tasks are still under investigation. Their solving methods are often researched independently of each other due to complexity of the tasks. As a result, systems that realize these methods turn out to be incompatible and therefore could not be used together in the lifecycle of a KBS. The following problem arises here: how to support the full life cycle of a KBS. This paper introduces a class of computer knowledge banks that are intended to support the full life cycle of KBSs. Primary tasks that arise in the full life cycle of a KBS are analyzed. The architecture of a knowledge bank of the introduced class is presented, including an Information Content, a Shell of the Information Content and a Software Content. General requirements on these components are formulated on the basis of the analysis. These requirements depend on the current state of understanding in the life cycle of KBSs.

Keywords:

Bank of knowledge; Knowledge based system; Information resource; Life cycle; Ontology

Introduction

Different kinds of information are used when solving tasks that arise in the life cycle of applied KBSs. Solving methods for these tasks are often researched independently of each other due to their complexity. Researchers develop prototypes of knowledge-processing systems to test the methods. As a rule, the problem of compatibility of the prototypes is not considered. For that reason, the representation of used information depends on a specific system in the most cases. Resulting systems turn out to be incompatible and therefore could not be used together in the life cycle of a KBS. The following problem arises here: how to support the full life cycle of a KBS.

A number of attempts have been made to develop compatible computer systems for supporting a process of solving bundled tasks in the field of medicine, mechanical engineering, commerce and also for heterogeneous and distributed information sources processing. Some of these systems [5, 13, 15] are oriented to support a process of solving bundled tasks in special domains. Systems like these usually contain a set of compatible computer tools intended to solve a number of applied tasks and integrate all the necessary information. Other systems [1, 12, 16, 17] provide facilities that support access to heterogeneous and distributed information sources in a way that isolates users from differences in the formats, locations and facilities of those sources. Systems like these are not restricted to any special domain, thus pretend to high universality. However, the problem of how to support the full life cycle of KBSs is not considered while developing mentioned systems. As a result, none of these systems could be used to support the full life cycle of any KBS.

The purpose of this paper is to introduce a class of information resources that are intended to support the full life cycle of KBSs. The paper is structured as follows. A notion of a computer bank of knowledge is advanced. An analysis of the primary tasks that are solved in the life cycle of KBSs is presented. Then requirements on an information content, on the means for access to the information content, and on a software content of the knowledge bank are stated on the basis of the analysis.

A notion of a computer bank of knowledge

A notion of a computer bank of knowledge is introduced to support coordinated researches for the methods of solving the tasks, which arise in the life cycle of applied KBSs. The bank of knowledge (knowledge bank, KBk) is an information resource that integrates all the peaces of information useful in the life cycle of applied KBSs, provides an access to them for computer programs and provides users with intelligent tools to cope with the tasks

having proven methods of solution.

The knowledge bank consists of an Information Content (IC), a Shell of the Information Content (SIC) and a Software Content (SC). The IC contains all the peaces of information useful in the life cycle of applied KBSs. The SIC provides computer programs with means for access to the IC. The editing tools are required to create and change the IC. At present, there are proven methods of editing various kinds of information [3, 4, 8, 14]. Thus, editing tools are a part of the SC. Other tools can be included into the SC as the methods for solving appropriate tasks are carried out.

At the initial stage, knowledge banks can be developed in order to support the life cycle of a concrete knowledge-processing system. The common purpose is to develop a multi-purpose knowledge bank. To achieve this purpose the following principle of aggregation is proposed: to develop this KBk is to aggregate all the available KBks. To aggregate a KBk_1 and KBk_2 into a common KBk is to aggregate the IC_1 and IC_2, SIC_1 and SIC_2, SC_1 and SC_2. To aggregate the IC_1 and IC_2 into the common IC is to integrate source descriptions and to resolve conflicts (e.g. domain, naming, structural, identification conflicts). To aggregate the SIC_1 and SIC_2 into the common SIC is to share their facilities between the KBk_1 and KBk_2. Note that the common SIC provides access both to the aggregated and distributed information sources. To aggregate the SC_1 and SC_2 into the common SC is to share their tools between the KBk_1 and KBk_2. To support the life cycle of any applied KBS is simply to retrieve all necessary pieces of information from the ICs of the multi-purpose KBk by means of the common SIC.

An analysis of primary tasks that are solved in the life cycle of KBSs

Applied tasks

The main objective of developing an applied KBS is to support people persons in solving applied tasks in a domain. To solve applied tasks in domain the KBS uses its knowledge base. For the computer to utilize knowledge, it must be expressed formally. Functional and predicative relations (mathematical logic) are usually used to formalize properties of empirical instances and relations between them. There are extensional (the tabular representation) and intensional (by formulas) ways to define the meaning of a relation. A set of meanings of terms for knowledge description (the tabular representation of atomic expressions) is a widespread special case. In addition, the meaning of such a term can also be represented by a formula. Demonstrative examples are knowledge in physics, chemistry.

The problem of knowledge editing

Domain knowledge is concrete definitions of meanings for the set of terms for knowledge description contained in the domain ontology [3, 8, 14]. Various groups of experts can uphold different domain ontologies. The domain ontology is the matter of their belief [10, 18], therefore it is impossible to reject any of proposed ontologies. Besides, the necessity of information about the observable reality (so called data ontology or base of observations) is substantiated in [3, 8]. The fact that the data ontology should be detached from the domain ontology is also substantiated in that paper. Various groups of experts can have different standpoints concerning the knowledge of a domain. They would form different knowledge bases in that case. But often it is difficult to determine whether one knowledge base is better (to some criteria) than other, i.e. a knowledge base can succeed in solving one problem and fail in other.

The problem of inductive forming knowledge bases

Computer methods for inductive forming of knowledge bases are elaborated. As a matter of fact, inductive forming is based on empirical data. A base of empirical data representing observed facts of the experience will be called a base of positive instances (PI-base). The PI-base should represent the learning set having special characteristics in addition to quantity of instances in order to obtain knowledge of higher quality. For example, it is important to have instances of occasional cases in the base of positive instances in order to obtain knowledge of high correctness. The problem of how to obtain the PI-base representing the best learning set arises here. A usual way to solve the problem is as follows: one has to form a PI-base, to generate inductively a knowledge base using this base of instances and then to compare the knowledge base with those of the previously generated using other bases of positive instances.

The PI-base where each instance is fully described is enough to solve the problem of inductive forming the best knowledge base. But if a number of the samples have not some information represented then the amount of corresponding knowledge bases turns out to be large. Additional information is needed to limit the number of knowledge bases. Such additional information is usually the base containing samples of unreal situations – so called base of negative instances (NI-base). Moreover, an adjacent task to the task of inductive forming knowledge base using the base of positive instances can be set. It is the task of inductive forming knowledge base using the NI-base where each negative instance is fully described. Each of the competing PI-bases can has a complementary NI-base corresponding to the same ontology.

Using domain ontologies when solving the problem of inductive forming knowledge base advances the clarity of formed knowledge for domain specialists [10].

The problem of debugging knowledge bases

The correctness of knowledge base, its high accuracy and corroboration with a large amount of empirical data are the requisite conditions of the applicability of any KBS using it. Meanwhile, a knowledge base obtained from an expert can miss these conditions. Namely, it can has the following defects: incorrectness, inaccuracy and poor corroboration. The problem of debugging knowledge base is in the successive defect elimination from it. Including an expert into this process is intended to reduce its calculating

complexity. In order to an expert can effectively control the process, he must be provided with an intellectual support. This problem can be solved using a domain ontology. The basic way to verify defects mentioned above is estimating a knowledge base against empirical and unreal data, which are in a PI-base and a NI-base.

The problem of ontology editing

Ontology is a kind of knowledge. For a program system can use an ontology, the ontology must be formalized and inputted into the computer. Forming an ontology is a complex task. This task needs an intellectual support. A promising approach to intellectual support for the task of information editing is suggested in [2-4, 8]. This approach requires a hierarchy of metaontologies to obtain and edit the desired ontology.

Special ontology description languages are used for formal representation of ontologies nowadays. Many ontology description languages were suggested till now [6, 7, 9, 11, 18], but none of them is generally accepted. So, different special languages can be used to formalize different ontologies (but single language is used to formalize a certain ontology).

The problem of editing bases of positive and negative instances

It is known that empirical data is observed on the basis of some a priori knowledge about the reality. A data ontology (base of observations) can be used as this knowledge. Taking into account the definition of NI-base, the data ontology can be used to provide the process of NI-base editing with an intellectual support as well. The data ontology combined with the corresponding PI-base can also be used to automatic generating a corresponding NI-base. A positive or negative instance is a set of concrete definitions of terms representing concepts of the domain. There are domains where accounting information accompanies each instance in PI-base. For example, the name of a patient, the date of an inspection, the name of a physician and so on accompany results of the patient examination in medicine. Problems that are solved using the base of accounting information (namely, statistical analysis and so on) are usually not concerned with the life cycle of KBSs. Thus, there is no need to include accounting information into the PI-base. A corresponding ontology (different from the data ontology) is needed to organize the process of accounting information editing. As it is indicated above, one of the possible aims of the PI-base editing task is to obtain a PI-base representing the best learning set. The possibility to store a number of competing PI-bases in the ICKBk is required to tackle the problem of testing PI-base against this criterion.

The problem of automatic generating a method for solving an applied task

Means for automatic generating methods for solving an applied task in a domain are worked out. A domain ontology and a mathematical specification of the task in terms of this ontology are used to do it [10].

The problem of reusing different kinds of knowledge

The practice shows us that the same ontologies and knowledge bases are often useful in different applications. That is, there are examples of outwardly different domains having the same ontology. For instance, sociology, psychology, biology, medicine and etc. all have so called causal ontology. The possibility to use a number of ontologies is required in order to support the reusing of mentioned ontologies.

There are domains where each section of a domain has an independent significance but all of them have the same ontology. From here it can be assumed that the domain ontology is reused in its sections. Medicine is an indicative example. Since parts of such an ontology have an independent significance, corresponding knowledge bases are independent (not competing) as well. There are also domains where sections have not their own significance. To meet this case the possibility to use a number of competing knowledge bases is required.

Requirements on a computer bank of knowledge

Requirements on the Information Content of a KBk

The Information Content of a KBk (ICKBk) is intended to contain all peaces of information useful in the life cycle of applied KBSs. The following requirements on the IC are deduced from the previous analysis.

The possibility to store a number of not competing, competing (corresponding the same problem domain) ontologies, and those of described in different formal languages in the ICKBk is required.

The possibility to store a number of not competing, competing (corresponding the same domain ontology) knowledge bases, those of represented by formulas, and those of represented as a set of meanings of terms (where the meaning of a term can also be represented by a formula) in the ICKBk is required.

The possibility to store a number of not competing and competing (corresponding the same data ontology) PI-bases as well as NI-bases in the ICKBk is required.

The possibility to accumulate ontologies, knowledge bases, PI–bases and NI–bases for a diverse of domains simultaneously is required.

The human element acts on a result of editing information. This process may result in a situation when it is easier to restore a previous state of information (having a number of changes revoked at once) rather than to roll the changes back step-by-step. Hence, the possibility to store a number of versions of each component in the ICKBk is required.

The progress in the field of supporting the KBS life cycle may result in changing present-day notions of ontology, knowledge base, PI-base and NI-base. Lets consider basic principles that are expected to be permanent through the

progress.

KBSs are based on knowledge. The (meta)ontology is assumed to be the most common kind of knowledge. There are some approaches to the definition of ontologies [3, 6, 7, 9, 10, 14, 18]. The following basic principles of construction of an ontology are deduced from that definitions.

1. An ontology should be a set of agreements (on terms and their possible meanings).
2. An appropriate formal language should be used to define an ontology.
3. Formal elements of an ontology should be accompanied by an informal description for non-mathematicians.
4. An ontology may depends on metaontologies but must not depend on more specific kinds of knowledge, such as knowledge base or base of real situations.

This set of principles is to retain the substance of the concept "ontology". A notion of ontology may change in all other respects. For instance, multimedia data ontologies can be organized as an alternative to the traditional (and mentioned above) data ontologies (bases of observations). Such data ontologies contain multimedia objects in addition to formalisms, e.g. a geographical map with a set of geographical coordinates of its objects. All other kinds of information that are in an ICKBk are understood as concrete definitions of meanings for corresponding set of terms contained in corresponding ontology. In view of this approach, structures of knowledge base, PI-base and NI-base are fixed by corresponding ontology. As for their content, there are tasks and kinds of information related to them that could turn out to be important in the life cycle of KBSs. For instance, multimedia bases are organized as an alternative to the traditional (and mentioned above) PI-bases.

However, such tasks and kinds of information are still under investigation. Being these kinds of information investigated, it is naturally to store them in the ICKBk. Thus, the possibility to support ontologies, knowledge bases, PI-bases and NI-bases in concordance with the progress (taking into account principles stated above) is required.

Requirements on the Shell of the Information Content of a KBk

The Shell of the Information Content (SIC) of the KBk is intended to provide computer programs with a function set for access to the ICKBk. Since the variety of tasks that might be solved using the ICKBk is not restricted, the completeness (sufficiency for solving any task using the ICKBk) of the access function set of the SIC is required.

Developers of knowledge-processing systems are people persons. Hence, the commonality of the access function set (the outward similarity of functions used to access to different components of the ICKBk) is required in order to ease a programming process. In addition, names of the functions must correspond with their functionality.

As is shown above, the ICKBk must provide the possibility to store ontologies described in different formal languages. An ontology is a set of agreements where each agreement is a proposition in a special formal language, i.e. mathematical formula. Moreover, it is shown [11] that an ontology description language (ODL) that is intended for formalizing ideas has to provide an extensibility. Therefore the SIC must provide the possibility to define extensible ODLs formally. Further, the SIC must provide the possibility to define any element of an ontology in the previously defined ODL. Since the variety of tasks that are solved using an ontology is not restricted, the SIC must provide an access to any defined element of an ontology to perform basic operations (usually getting, changing or deleting).

Since the ICKBk must provide the possibility to store knowledge bases represented by formulas, the SIC must provide the possibility to process knowledge bases represented by formulas. Since the ICKBk must provide the possibility to store knowledge bases represented by a set of meanings of terms, the SIC must provide the possibility to process knowledge bases represented by a set of meanings of terms.

The way of storing empirical data (which are in a PI-base of the ICKBk) in a database (usually relational) is widely accepted. A NI-base can have the same representation. Thus, the SIC must provide the possibility to access to a PI-base and NI-base as if they are databases.

Since the ICKBk must provide the possibility to store versions of its components, the SIC must provide the possibility to manage versions of those components.

Groups of researchers, and also KBS and supporting tool developers usually have remote location. The Internet network is an efficient means of communication for them. Therefore the SIC must provide the possibility to access to distributed (e.g. in the Internet) information sources having supported format.

Since collection of kinds of information in the ICKBk can be expanded, the possibility to expand the function set of the SIC is required in order to provide an access to new kinds of information.

Requirements on tools for editing the Information Content of a KBk

That the ICKBk must contain ontologies, knowledge bases, PI-bases and NI-bases is shown above. Special editing tools are used to create and edit those components of the ICKBk.

Requirements on tools for editing the ICKBk

Users of editing tools are people. Hence, an editing tool must provide its user with an intellectual support satisfying his or her needs for editing the ICKBk. This intellectual support includes providing user with the following information: what data is needed to the editing tool for current step and which answers are possible. It is also includes (if possible) preventing or verification of formal errors in user's answers.

The commonality of ways for editing the ICKBk (the similarity of ways used to edit different components of the

ICKBk) is required in order to minimize users' education costs and maximize users' productivity.

An approach suggested in [2-4, 8] could be used to edit ontologies, knowledge bases, PI-bases and NI-bases. Let us consider a process of editing the ICKBk using this approach.

An ODL is needed to define an ontology. An ontology of an ODL must be defined for the first. It is a complex task, and it is not concerned with any domain. The editing ontology task is in editing formulas. Hence, the skill in mathematics is needed to tackle this task. A knowledge engineer is called to tackle it therefore. To define the ontology of an ODL a tool intended for editing formulas having the following properties is required: (1) providing the formality of the process, (2) using the mathematical notation, (3) providing, if possible, an automation of computations and (4) preventing or verifying formal errors in edited information.

It is possible to organize an intellectual support for the knowledge engineer in the process of forming and editing an ODL. This process is in the defining the means of the terms describing language structures containing in the ODL ontology. Therefore, a tool intended for concrete definition of meaning of the terms contained in the ODL ontology is required for a knowledge engineer to define the ODL. As is already indicated, an ODL that is intended for formalizing ideas has to provide an extensibility. A possible structure of such a language is suggested in [11]. Namely, a class of languages consisting of a kernel and of special extensions is introduced there. In the case of these extensible languages, to define an ontology of an ODL is to define an ontology of an ODL kernel and ontologies of all its extensions. To use an ontology of an ODL is to use mentioned ontologies in the aggregate. Therefore, any tool using an ontology of an ODL must be capable to use a number of existing ontologies in the aggregate, i.e. as a single ontology.

The next common step is forming a hierarchy of metaontologies. This task is in the investigating and describing common properties of domains, e.g. causality [8], and requires special skills. A knowledge engineer is called to tackle it therefore. *A tool intended for editing formulas having properties 1, 2, 3 and 4 is required* again in order to define a metaontology.

Provided that a metaontology is defined, it is possible to organize an intellectual support for the knowledge engineer in the process of forming and editing an ontology. This process is in defining the meanings of special terms containing in the metaontology. Therefore, a tool intended for defining the meanings of the terms of the metaontology is required for a knowledge engineer to define an ontology. Since ontologies can be reused, any tool using an ontology must be capable to use a number of existing ontologies in the aggregate, i.e. as a single ontology. As is indicated in [2], there are situations when a domain ontology must be formed and edited by a domain expert directly. An expert oriented intellectual support is needed to meet this case. More exhaustive metaontology can be used to organize an expert oriented intellectual support [2, 8].

An abstract ontology can be made more exact. To make an ontology more exact is to add a number of specific propositions (i.e. formulas) to it and to declare some more special terms – i.e. it is to edit the ontology. *A tool intended for editing formulas having properties 1, 2, 3 and 4 is required* again.

A domain knowledge-editing problem is in defining the meanings of the set of terms for knowledge description contained in a domain ontology. Domain experts have necessary skills to tackle this problem, but usually they are not competent in mathematics and dealing with a computer. A domain expert needs the following intellectual support therefore: he needs to use terms of his domain and to direct his efforts toward solving usual tasks. Hence, a tool intended for defining the meanings of the terms for knowledge description, which provides using expert terminology and directs expert's efforts toward solving usual tasks, is required for a domain expert to edit a knowledge base.

The empirical data (i.e. a PI-base) editing problem is in defining the meanings of the terms for data description contained in a data ontology. Domain specialists have necessary skills to tackle this problem, but usually they are not competent in mathematics and dealing with a computer too. Therefore, a domain specialist needs the following intellectual support: he needs to use terms of the domain (i.e. expert terminology) and to direct his efforts toward solving task of observation and registration facts. Hence, a tool intended for defining the meanings of the terms for knowledge description, which providing using expert terminology and directs specialist's efforts toward solving the task of observation and registration facts, is required for a domain specialist to edit a PI-base.

It is recognized nowadays [2-4, 8, 14] that the orientation of the intellectual support toward a domain expert or domain specialist is ontology-driven.

The need to store versions of information components is reasoned above. Therefore, editing tools must provide the possibility to store and use different versions of information components.

There are situations when a user of an editing tool has a need to gather pieces of prepared information and to output them in a human-readable form. For instance, there is a necessity to output a whole ontology or knowledge base in a human-readable form in order to discuss obtained results. A knowledge base in a human-readable form can also be used for educational purposes. Descriptions of real situations can be used for many purposes, e.g. statistical, bookkeeping and so on. Thereby the necessity of generating reports is clear. A report is a list of pieces of information (namely elements of an ontology, a knowledge base, a PI-base or NI-base) represented in a human-readable form. Hence, editing tools must provide the possibility to generate reports from information components of the ICKBk.

Comments on requirements on tools for editing the ICKBk

It follows from the set of requirements on tools for editing the ICKBk that the approach suggested in this work differs from the one described in [2, 6, 7, 9, 18] and widely accepted.

Namely, the widely accepted approach is the following. A computer ODL is proposed. An ontology is formed and edited as a text in that language. The text of the ontology can be printed in order to discuss it. To utilize this ontology it has to be compiled. In contrast to widely accepted ontology editing tools, knowledge base (PI-base) editing tools are specialized in an intellectual support of domain experts (specialists). This intellectual support commonly is in the questioning an expert (specialist) about necessary information, providing him with the set of possible answers and (if possible) preventing or verifying formal errors.

Widely accepted approach has the following shortcomings. A knowledge engineer has to distribute his efforts among learning special computer ODLs, forming an ontology by text editors and ensuring the syntactic and semantic validity of the ontology. None of the domain experts is able to form any domain ontology in an ODL. What is more, compiling routines for ODLs are needed.

Meanwhile, each formal ODL has an ontology underlying it. Moreover, it is recognized that ontologies must have the declarative semantics in contrast to algorithmic computer programs having the procedural semantics. Thereby there is no need to form or edit an ontology as a sequential text. Ontology propositions can be formed or edited independently of each other (but by the necessary intellectual support). A principle of (meta)ontology acquisition and editing by analogy with that of knowledge base and PI-base acquisition and editing is suggested instead of that of ontology text description using special computer languages. This principle comes out from the analysis of the present-day situation in the life cycle of KBSs. It is to eliminate widely accepted approach shortcomings and it takes last remarks into account. It has the following advances:

1. There is no more need for a knowledge engineer to learn special computer ODLs.
2. A knowledge engineer concentrates all his efforts on the problem of forming an ontology.
3. Computer programs ensure the syntactic validity of an ontology being edited instead of knowledge engineers.
4. Given a metaontology of a domain, a domain expert can form an ontology of the domain.
5. There is no more need to develop compiling routines for ODLs.
6. The possibility to produce human-readable reports on an ontology is provided by special means.

Note that interfaces of editing tools instead of computer ODLs must be standardized when using the suggested approach.

Extensions of the Software Content of a KBk

The Software Content of the KBk (SCKBk) is intended for containing all the software systems that are useful or necessary in the life cycle of KBSs. There are noteworthy problems (and kinds of information and software systems related to them) that might prove their significance in the life cycle of KBSs. For instance, methods for inductive forming and debugging knowledge bases are investigated. High-level information query languages are proposed. Also, as it indicated above, problems of obtaining formal knowledge from full-text information sources and using multimedia bases are investigated. However, such problems are still under investigation. Being they are carried out, it is naturally to include corresponding software systems into the SCKBk. At present there are only editing tools in the SCKBk. Thus, the possibility to expand the collection of the software systems in the SCKBk is required.

Conclusion

This paper has presented an analysis of the KBS life cycle support problem. A notion of a special information resource class has been introduced in order to solve the problem. An information resource of the introduced class has been called a bank of knowledge. A set of requirements on the bank of knowledge has been considered. These requirements depend on the current state of understanding in the life cycle of KBSs.

Acknowledgments

This work was carried out with financial support of the Russian Fund of Fundamental Investigations (grant 99-01-00634).

References

[1] Bayardo R., Bohrer W., Brice R., Cichocki A., Fowler G., Helai A., Kashyap V., Ksiezyk T., Martin G., Nodine M., Rashid M., Rusinkiewicz M., Shea R., Unnikrishnan C., Unruh A., Woelk D. 1997. InfoSleuth: agent-based semantic integration of information in open and dynamic environments. In Proceedings of the ACM International Conference on the Management of Data (SIGMOD), pp. 195–206. Tucson, Arizona.

[2] Eriksson H., et al. 1999. Automatic Generation of Ontology Editors. In Proceedings of the Twelfth Banff Knowledge Acquisition for Knowledge-based systems Workshop, Banff, Alberta, Canada.

[3] Eriksson H. 1994. Models for knowledge-acquisition tool design. *Knowledge Acquisition* 6: 47-74.

[4] Eriksson H., Puerta A. R., Musen M. A. 1994. Generation of knowledge-acquisition tools from domain ontologies. *International Journal of Human and Computer Studies* 41: 425-453.

[5] Firestone J. M. DKMS Brief No. One: The Corporate Information Factory or the Corporate Knowledge Factory? See http://www.dkms.com/white_papers.htm.

[6] Genesereth, M.R., Fikes R.E. et al. 1992. Knowledge

Interchange Format (version 3.0) Reference Manual, Technical Report, Report Logic-92-1. Interlingua Working Group of the DARPA Knowledge Sharing Effort, Computer Science Department, Stanford Univ.

[7] Gruber T.R. 1992. Ontolingua: A Mechanism to Support Portable Ontologies. Technical report, KSL-91-66, Knowledge Systems Laboratory, Stanford Univ.

[8] Gruk A.V., Kleshchev A.S. 2000. Tools for intellectual supporting the acquisition process of different kinds of knowledge. A model of the process. Technical report 23-2000, IACP, Russian Academy of Sciences. See http://www.iacp.dvo.ru/es/ (in Russian, annotated in English).

[9] van Heijst G., Schreiber A.Th., Wielinga B.J. 1996. Using Explicit Ontologies in KBS Development. *International Journal of Human and Computer Studies* 46 (2-3): 183-292.

[10] Kleshchev A.S., Artemjeva I.L. 1999. Domain ontologies and knowledge processing. Technical Report 7-99, IACP , Russian Academy of Sciences. See http://www.iacp.dvo.ru/es/.

[11] Kleshchev A.S., Artemjeva I.L. 2000. Unenriched Logical Relationship Systems. Technical Report 1-2000, IACP , Russian Academy of Sciences. See http://www.iacp.dvo.ru/es/.

[12] Mena E., Illarramendi A., Kashyap V., Sheth A.P. 2000. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Distributed and Parallel Databases* 8: 223-271.

[13] Muhin A.V., Spiridonov O.V. 1997. Some approaches to developing computerized information support systems for process design and production control. See http://asutp.h1.ru/st2h.shtml (in Russian, annotated in English).

[14] Musen M. A., Fagan L. M., Combs D. M., & Shortliffe E. H. 1988. Use of a domain model to drive an interactive knowledge editing tool. *Knowledge-Based Systems*, Volume 2 (Knowledge Acquisition Tools for Expert Systems), London: Academic Press.

[15] Sterling W. 1998. The National Medical Knowledge Bank. In Proceedings of the 24th VLDB Conference, New York City.

[16] Singh M., Cannata P., Huhns M., Jacobs N., Ksiezyk T., Ong K., Sheth A., Tomlinson C., Woelk D. 1997. The Carnot heterogeneous database project: implemented applications. *Distributed and Parallel Databases* 5: 207-225.

[17] Smith G. Unified Information Access System. 2001. Status report: April 2001. See http://uias.calstate.edu/uias_status.htm.

[18] Uschold M. 1998. Knowledge level modeling: concepts and terminology. *The Knowledge Engineering Review* 13/1: 5-29.