Bull. Nov. Comp. Center, Comp. Science, 31 (2010), 195–207
© 2010 NCC Publisher

An approach to development of the decision support system for enterprise with complex technological infrastructure

Yu. A. Zagorulko, G. B. Zagorulko

Abstract. The paper presents an approach to development of a system that supports decision-making for tasks aimed to reduce power consumption and enhance the environmental safety of a large-scale enterprise with a complex technological infrastructure. The architecture and operating principles of this system are discussed. Adjustment of the system to the subject domain and types of tasks to be solved is provided by including explicitly the models of the subject and problem domains represented by ontologies into the system.

Keywords. Decision support system, task ontology, subject domain ontology, solver, decision support module, monitoring, analysis, optimization

1. Introduction

At present, the problem of increasing the power efficiency and environmental safety of large-scale enterprises with a complex technological infrastructure (ECTI) is of vital importance. A great number of automatic control systems are now exploited at such enterprises. But these systems, as a rule, are installed at local objects and serve only the main production process, not including the transportation services and the scheduled and emergency repair. In addition, these systems provide control and monitoring only of a single object or process, whereas effective decision making requires the integral information about all objects and processes of the enterprise technological infrastructure. Thereby such systems cannot provide a decision-maker with information about all processes which influence the efficiency and ecological safety of the enterprise operation.

To reduce the power consumption of ECTI and enhance the environmental safety of its operation, the system of operational monitoring of its technological infrastructure (SOMTI) is required. An important component of this system is a decision support system (DSS) [1] intended to solve the following tasks:

- Analysis of the state of objects of ECTI technological infrastructure (further – the objects) for the purpose of making advices on its operation improvement and accident prevention.

- Analysis of static and dynamic information about the objects for the purpose of elaboration of suggestions for a decision-maker on scheduled service and/or emergency repair of the objects, as well as their decommissioning and substituting with new objects instead.
- Elaboration of suggestions for a decision-maker on optimization of traffic flows and processes of maintenance and repair of the objects.

Technological infrastructure of the enterprise can undergo both structural and qualitative changes. For example, new kinds of equipment can be installed and decision of new tasks may be required. Therefore the system of operational monitoring of the ECTI technological infrastructure, as a whole, and DSS, in particular, must be adjustable to the subject domain and types of tasks. So, DSS is designed in such a way that the subject domain model is included in DSS and its architecture allows us to include additional modules that provide decision of new tasks.

The architecture flexibility is substantially provided by the use of ontology [2, 3] as a universal format for data representation in the system, which allows one to unify and considerably simplify information exchange between heterogeneous components and modules of the DSS and SOMTI.

Recently, ontologies are quite frequently used in DSS intended for decision-making in various areas: medicine [4], law (justice) [5], industry [6], ecology [7], etc. Thus the ontology-driven DSS are applied in clinical management [4, 8], management system audit [9], management in metal industry [6], security management in heterogeneous networks [10], wastewater management [7], in improving data warehouse performances [11], etc.

The paper discusses the architecture, design and operating principles of a DSS intended for decision-making in the system of operational monitoring of ECTI technological infrastructure, as well as the classes of tasks to be solved.

2. Architecture of DSS SOMTI

When the DSS elaborates suggestions for a decision-maker, it uses information from the central data warehouse (CDW) of SOMTI, where data about the topology and state of technological infrastructure of the enterprise are presented. Due to this fact, DSS is implemented as two interacting modules – the DSS adapter and a supervisor which organizes the operation of solvers providing decisions of a certain class of tasks in DSS (see Figure 1).

Decisions of concrete SOMTI tasks are made by special decision support modules which are performed (interpreted) by one of the solvers. Each solver has its own format of input and output data, therefore, for each of them, adapters for data exchange with the DSS local memory are developed. These adapters perform additional transformations of input data in order to

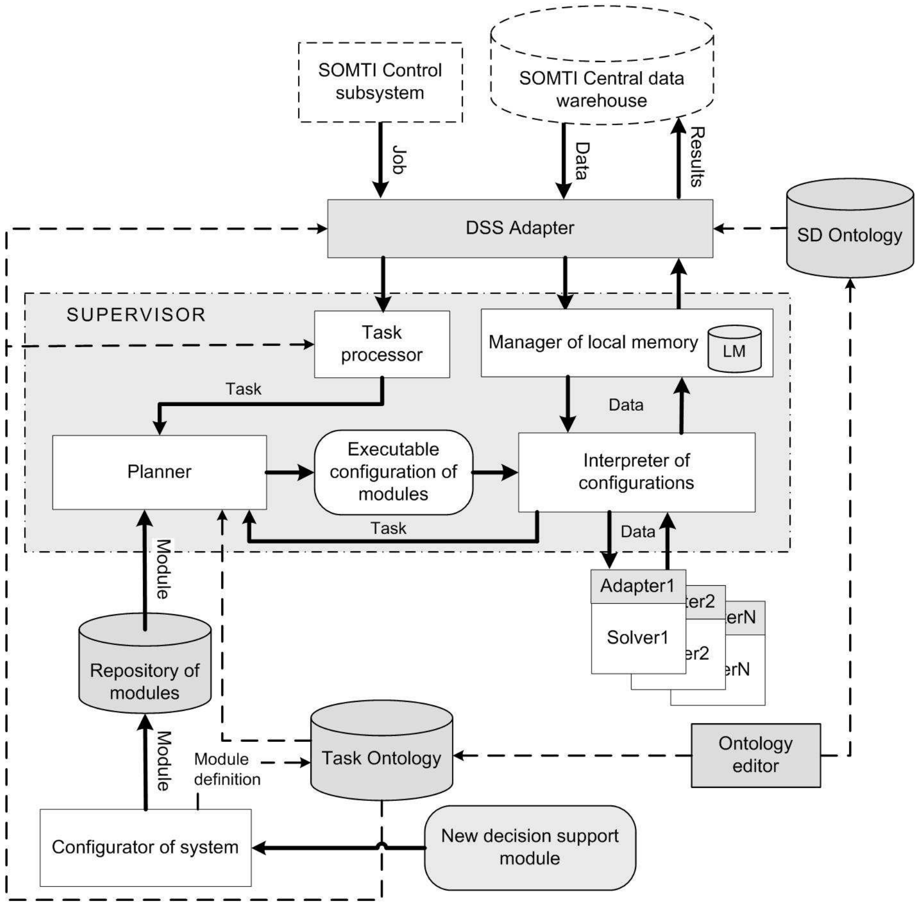

Figure 1. The DSS architecture

adjust them in accordance with solver formats, and after solvers execution they interpret the output data (results) converting them into the internal representation used in the DSS.

In order to simplify and unify information exchange between heterogeneous components and modules of the DSS (adapters, supervisor, solvers, modules of decision support etc.) and CDW SOMTI, we have developed a data representation format in the form of concepts instances of ontology describing the subject and problem domains of the system. This ontology consists of two interdependent ontologies – subject domain ontology and task ontology [3, 12].

In addition, the DSS includes a set of auxiliary tools (ontology editor and configurator of the system) that support adjustment and configuration management of the system.

The supervisor organizes task processing, preparing input data for solvers, collecting computation results and sending them to CDW SOMTI. The supervisor consists of the following modules: task processor, planner, interpreter of configurations, and manager of local memory.

The DSS adapter provides task receiving from the control subsystem (CS) SOMTI, data uploading and downloading from CDW, as well as communication with the supervisor.

Communication of the DSS adapter with CDW SOMTI is performed through a special program interface which provides the methods for access to objects (data) from CDW and for creation of objects that are the results of task solution and recording them to CDW.

The DSS adapter interaction with SOMTI control system is implemented in a form of a message system.

Communication of the DSS adapter with the supervisor consists in interpretation of objects received from CDW in terms of the ontology and transferring them to the supervisor, and reversely, interpretation of output data received from the supervisor and transferring them to CDW.

The operation of DSS SOMTI will be described in detail in section 3.

3. Knowledge representation in DSS SOMTI

As was said above, the system ontology consists of the subject domain ontology (SD ontology) and the task ontology.

The subject domain ontology (SD ontology) describes the model of a subject domain in the form of concepts (classes) and relations between them. The basic classes of SD ontology are *Object, Normative-reference object, State*, and *Results* (see Figure 2).

The class *Object* includes the following subclasses: *Equipment*, *Mobile objects* and *Road network.* Objects can be in a state described by the class *State.* In turn, it includes the following subclasses: *Disrepair, Breakdown, Normal state, Emergency state* etc.

Description of topology of the enterprise technological infrastructure is an important part of SD ontology. The positional relationship and connectivity of objects of the technological infrastructure are defined by relations “Contain”, “Enter”, “Exit”.

To control the correspondence of the analyzed objects parameters to their standard values, the normative-reference information is used. So, for each concept describing the type of equipment or a container of equipment, a special concept named *NR-object* (a normative-reference object) is introduced. *NR-object* has the same set of attributes as the original concept (object), but the numeric values of *NR-object* attributes are given as a pair of numbers which define the variation interval of attributes of the original concept according to technical standards.

Each object of the technological infrastructure has its *Repair schedule*,

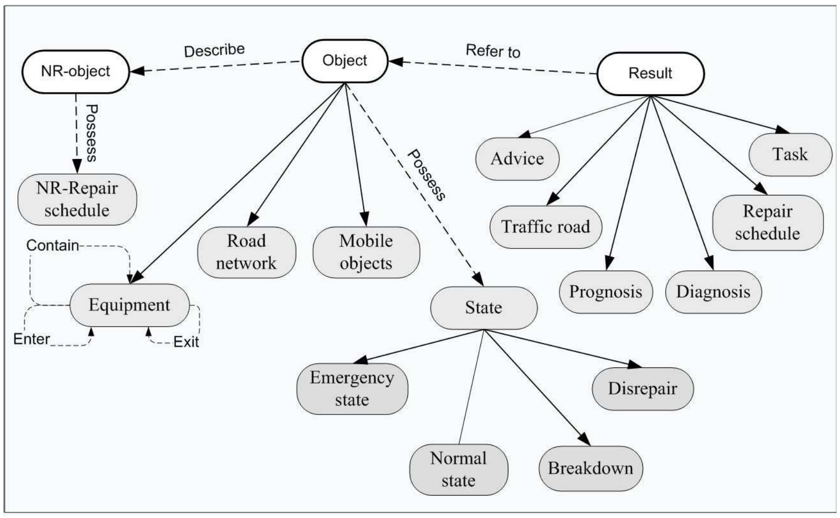

Figure 2. The subject domain ontology

where information about scheduled, current and urgent repairs of this object is presented.

The *Result* of the DSS operation can be *Diagnosis* of the state of equipment or mobile object, *Advice* for a decision-maker, *Prognosis* of changes in the object state, solution of the transport task (*Traffic road*), corrected *Repair schedule*, etc. A task spawned by the decision support module which performs the detailed analysis of an object can also be delivered as a result of the DSS operation.

The task ontology includes the description of tasks which are solved by the system and the decision support modules performing these tasks. Relations between tasks and between tasks and decision support modules are introduced into the ontology.

For a task description, a concept (class) *Task* with the attributes "Name of task" and "Parameters of task" is used.

The attribute "Parameters of task" includes a set of pairs $\langle C_{SD}, L_O \rangle$, where C_{SD} is a class of the subject domain ontology, L_O is a list of names of objects of the class C_{SD} for which this task should be solved (it is supposed that the object's name is its key attribute, i.e. the object can be uniquely identified by its name). If L_O is not given (or empty), it is assumed that the task should be solved for all objects of C_{SD}.

To represent the decision support module, the class *Module* with the attribute "Name of module" is introduced into the task ontology.

The relations "Subtask" and "Spawn" are defined on tasks. The first relation connects a task with others (its subtasks) that should be solved in order to solve this one. The relation "Spawn" defines a potential possibility of generation of a task by another one. The spawned task can perform, for example, the detailed analysis (diagnostics) of one object or group objects, a hypothesis of malfunction of which was put forward by “the main task”. In this case, the name of the object is transmitted to the spawned task as one of its parameter. Note that the spawned task is considered by the system as a separate independent task and executed after termination of “the main task” (see Section 3).

For linking a task with the decision support module which provides a solution of this task, the relation “Realize” is introduced into the task ontology. It should be noted that a module can be linked only with a terminal task, i.e. a task which does not include subtasks.

All decision support modules are stored in a special repository and supplied with the following attributes: “Name of module”, “Input data”, “Output data”, and “Solver”.

The attribute “Name of module” gives the name of the decision support module.

The attribute “Input data” defines the set of types (classes) of objects which are necessary for the module operation. The value of this attribute has the form $\langle C_{IN}, R_{IN}, A_{IN}\rangle$, where C_{IN} is a set of classes of SD ontology, $R_{IN}$is a set of relations of SD ontology defined on the classes C_{IN}, A_{IN} is a set of constraints on the values of attributes of objects of classes from C_{IN}. The set of constraints A_{IN} serves for filtration of objects for which the task should be solved. If constraints for some class are not given, then all objects of this class will be taken from CDw. If a module performs a task with parameters, they are added to the set of constraints A_{IN}.

The attribute “Output data” specifies the set of classes whose objects can be created by the module operation. The content of this attribute has the form $\langle C_{OUT}, R_{OUT}\rangle$, where C_{OUT} is the set of classes of SD ontology, $R_{OUT}$is the set of relations of SD ontology defined on the classes C_{OUT}.

The attribute “Solver” defines the name of a program system which will execute (interpret) the decision support module.

To describe the ontology, we use the knowledge representation language of the Semp-TAO system [13, 14]. This language was selected because, firstly, it has a rich set of data types and allows one to represent the concepts of subject and problem domains in the form of classes of objects and relations, and secondly, it includes facilities which allow one to specify the decision-making procedures and information processing required in the DSS in the form of production rules working with the objects of specified classes. Due to this fact, the rules in decision support modules implemented in the production model paradigm can be described in terms of ontology.

4. Operating principles and adjustment of DSS SOMTI

The main cycle of DSS SOMTI operation consists in successive execution of jobs coming from the control system.

Each job contains a description of some task known to the DSS. (A task is considered as known to the DSS if its description is presented in the task ontology.) The job comes to the DSS adapter that checks the task described in the job against the task ontology, converts the task into the supervisor format and transmits it to the supervisor for further processing.

The task processor, which is a part of the supervisor, analyses the incoming task description, represents it in the task ontology format and transmits it to the planner.

The planner for each incoming task generates an executable configuration of modules which perform this task. During this process, the planner turns to the task ontology in order to know the name of the decision support module (in a general case – a group of modules) performing this task. Next, the planner gets the description of this module from the repository of the decision support modules and puts it in their configuration.

Operation of the interpreter of configurations consists in the following:

- selection of the next module from the configuration of modules,
- upload of the data necessary for its operation from CDW (through the DSS adapter) to local memory (LM),
- activation of the corresponding solver with simultaneous transmission of data from LM to it and to the selected module,
- download of the module operation results to LM (and to CDW if the module is the last one in the configuration of modules).

Note that data exchange between solvers and local memory is performed through adapters.

After processing of the next module, the supervisor carries out monitoring of LM and if it finds the objects of the class *Task*, then it forms them in a job and calls the planner. (It should be noted that any decision support module can spawn a new task represented by an instance of the class *Task* from the task ontology.)

After finishing the operation of the planner which can supplement the configuration with new module descriptions, the interpreter of configurations is activated and proceeds its operation with already modified configuration of modules.

After all tasks that came to the supervisor and all tasks “spawned’ by them are solved, the supervisor stops its operation and goes into the state of waiting for new tasks.

The technological infrastructure of an enterprise can undergo both structural and qualitative changes. For example, new kinds of equipment can be installed, which, in turn, can result in the necessity to decide new tasks or to modify previous decisions earlier implemented. Therefore, facilities for adjustment to the subject domain and types of tasks are included in the DSS.

So, new kinds of equipment coming into service at ECTI should be reflected, first of all, in the subject domain ontology. To this end, the corresponding classes of objects, as well as new types of relations which link new classes of objects with the classes already presented in the ontology, should be added to the ontology using the ontology editor (see Figure 1). In this case, we will possibly need to modify descriptions of some decision support modules, namely, to complement descriptions of their input and/or output data and modify or complement their sets of production rules.

When new tasks appear, they should be described and added to the task ontology. In addition, an individual decision support module should be implemented for each of these tasks. Then this module should be supplied with a description in the format presented in Section 2, and registered and included into the repository of modules with the help of the system configurator (see Figure 1). Additionally, the new module should be added to the task ontology and linked with “its” task by the relation “Realize”.

The DSS is designed in such a way that its architecture allows one to include not only additional modules that provide the decision of new tasks but new solvers also. So, if it is necessary to solve a new class of tasks, which are not solved by the solvers already included in the system, then a new solver can be included and registered in the DSS by means of the system configurator. After that the required decision support modules are implemented and added to the task ontology and the repository of modules as described above.

5. Decision-making support for oil-and-gas production enterprise (OGPE)

The suggested approach was used for development of the DSS which serves an oil-and-gas production enterprise (OGPE). This DSS supports decision of two kinds of tasks, analytical and optimization, therefore two solvers are exploited in it. These are Semp, an interpreter of production rules of the Semp-TAO system, and GLPK, the GNU Linear Programming Kit package [15] intended to solve the problems of large-scale linear programming, mixed integer programming, and other related problems. For each of these solvers, adapters for data exchange with the DSS local memory (the Semp adapter and GLPK adapter) have been developed.

The task ontology of the developed DSS is presented in Figure 3.

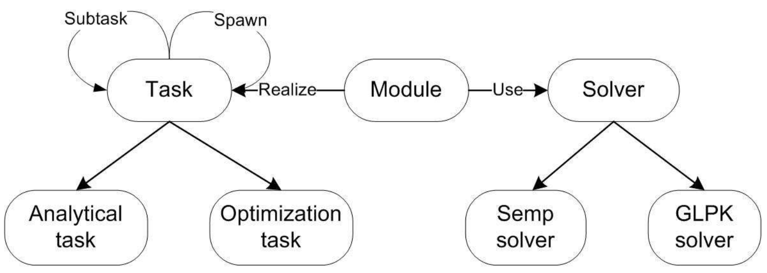

Figure 3. The task ontology of the DSS serving oil-and-gas production enterprise

In the DSS, two optimization tasks are solved: computation of optimal traffic routing for tank trucks which transport oil from step-out wells to the central oil collection site and optimization of the schedule of OGPE equipment repair. Decision support for these tasks is performed by the GLPK solver. In this solver, a model of a task solution, its input data and the solution results are represented in a special language GMPL (GNU MathProg Language). This language allows us to describe a task in the mathematical notation including parametric equations and inequalities, indexed expressions, etc., which considerably simplifies model building for experts.

In the DSS, the class of analytical tasks is represented more widely. They are the following:

- Decision-making under detection of a pipeline break.
- Decision-making under emergency on the transformer substation.
- General monitoring of the OGPE main performances.
- Monitoring of the well cluster.
- Equipment diagnostics.
- Decision-making under detection of imbalance of the liquid-gas mixture and the electric power.
- Analysis of the well cluster mode of behavior and making advises on determination of a new (safe/optimal) mode of behavior.
- Decision-making under imbalance of output of a well cluster and liquid injected into bed.

The subject domain ontology of the DSS was supplemented by concepts related to oil-and-gas production domain. In particular, the class *Equipment* was extended by such subclasses as *Pumping facilities, Transformers, Power transmission lines* etc. Subclasses corresponding to various kinds of auto transport, namely, *Tank trucks, Trucks, Buses* etc., were added to the class *Mobile objects.*

Virtually, all analytical tasks use the notion of a trend. By this we mean the tendency of changing some parameter of the analyzed OGPE object represented as a numeric characteristic. For example, pump efficiency, power consumption at the booster pump station, transformer capacity and others can be such a parameter. When time series of some characteristic are input data of a task, a special function on time series determines the trend in the form of one of the followng linguistic values: *sharp decrease, decrease, slight decrease, stability, fluctuation in the normative range, sharp increase, increase*, and *slight increase*. Using time series, one can detect the presence and moment of bursts and falls of the characteristics values.

Interdependency of characteristics trends and the equipment state is specified by production rules on the basis of expert knowledge. For example, if for some pump the electrical energy consumption increases and its efficiency and capacity decrease, and these parameters, being in the range of norm, tend to their boundaries, then it is highly probable that this pump will be broken.

To solve the analytical tasks, the decision support modules implemented in the production model paradigm are used. As was said above, the production rules are specified in the knowledge representation language of Semp-TAO [14] and the interpreter of this system is used for execution of these modules. They include a structured set of production rules, whose premises and corollaries are described in terms of the system ontology.

Let us give an example of a production rule monitoring the situation with a pump described above:

```
Forall nr: NR_Pump (Efficiency_min),
       p: Pump (Capacity, Energy_consumption, Efficiency),
       Describe (p, nr),
       Tendency (n.Capacity) = "decrease" &
       Tendency (n.Efficiency) = "decrease" &
       Tendency (n.Energy_consumption) = "increase"
=>
    Create Task (name: "Diagnostics_pump",
                     parameters: {< Pump, n >});
    cm := Critical_moment ( n.Efficiency, ns.Efficiency_min);
    Create Prognosis (message:
      "Efficiency will go at critical level in" + cm + "hours",
                  parameters: {< Pump, n >}).
```

6. Conclusions

In this paper, we present the principles of design and operation of the decision support system for a decision-maker whose aims are to reduce the power consumption and to enhance the environmental safety of a large-scale enterprise with a complex technological infrastructure. The model of the subject and problem domains represented by the subject domain and task ontologies is included into the DSS, which provides adjustment of the system to the subject domain and types of tasks. The architecture flexibility is provided by the use of ontology as a universal format for data and task representation in the system. It allows us to unify and considerably simplify information exchange between heterogeneous components and modules of the DSS and addition of new types of tasks to be solved.

The suggested approach was used for development of the initial version of the DSS for oil-and-gas production enterprises. It provides the decision support of tasks presented in section 4. Successful implementation of this system demonstrates productivity of the suggested approach to the DSS development on the basis of ontology.

Now we are working on the development of new decision support modules and their connection to the system. Besides, we discuss the possibility to include into the DSS one more solver – the UniCalc solver [16, 17] which we want to apply to solution of optimization tasks instead of or together with the GLPK solver. UniCalc was designed to solve arbitrary systems of algebraic and algebraic-logic relations, i.e., equations, inequalities and logical expressions. A system to be solved can be overdetermined or underdetermined, and the system's parameters (coefficients, variables, constants) may be given imprecisely, in the form of intervals. Such a system may contain both integer and real variables or a combination of them.

Acknowledgments. This work is performed within the framework of the Federal target program "Research and Developments Relating to the Priority Directions of Russia's Scientific and Technological Complex for 2007–2012" (the Government contract 02.514.11.4126 with the Russian Federal Agency for Science and Innovation).

References

[1] Turban E.,Jay E.A. Decision Support Systems and Intelligent Systems. 6^{th} ed. – Upper Saddle River, NJ.: Prentice Hall, 2001.

[2] Gruber T. Toward principles for the design of ontologies used for knowledge sharing // Internat. J. of Human-Computer Studies. – 1995. – Vol. 43, Iss. 5–6. – P. 907–928.

[3] Guarino N. Formal Ontology in Information Systems // Formal Ontology in Information Systems / N. Guarino (ed.). – Proc. of FOIS'98. – Trento, Italy: IOS Press. – P.3–15.

[4] Hussain S., Abidi S. R., Abidi S. S. R. Semantic Web Framework for Knowledge-Centric Clinical Decision Support Systems // Proc. of the 11-th

Conf. on Artificial Intelligence in Medicine, AIME 2007. – Lect. Notes in Artificial Intelligence. – 2007. – Vol. 4594 – P. 451–455.

[5] Casanovas P., Casellas N., Vallbe J.-J. An Ontology-Based Decision Support System for Judges // Proc. of the 2009 Conf. on Law, Ontologies and the Semantic Web: Channelling the Legal Information Flood. – Amsterdam: IOS Press, 2009. – P. 165–175.

[6] Li Sh.-T., Hsieh H.-Ch., Sun I-W. An Ontology-Based Knowledge Management System for the Metal Industry // Proc. of the Twelfth Internat. World Wide Web Conf. (WWW2003), Budapest, Hungary, 2003.

[7] Ceccaroni L., Cortés U., Sànchez-Marrè M. OntoWEDSS: augmenting environmental decision-support systems with ontologies // Environmental Modelling & Software. – 2004. – Vol. 19, Iss. 9. – P. 785–797.

[8] Colantonio S., Martinelli M., Moroni D. et al. A Decision Support System for Aiding Heart Failure Management // Proc. of Ninth Internat. Conf. on Intelligent Systems Design and Applications, Pisa, Italy, 2009. – Los Alamitos, CA, USA: IEEE Computer Society. – P. 351–356.

[9] Ishizu S., Gehrmann A., Minegishi J., Nagai Y. Ontology-Driven Decision Support Systems for Management System Audit // Proc. of the 52nd Annual Meeting of the ISSS, Madison, 2008.

[10] Choras M., Kozik R., Flizikowski A. et al. Ontology-Based Decision Support for Security Management in Heterogeneous Networks // Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence / Huang, D.-S. et al. (Eds.). – Proc. of 5th Internat. Conf. on Intelligent Computing, ICIC 2009, Ulsan, South Korea, 2009. – Lect. Notes in Artificial Intelligence. – Springer, 2009. – Vol. 5755.

[11] Nicolicin-Georgescu V., Benatier V., Lehn1 R., H. Briand An Ontology-Based Autonomic System for Improving Data Warehouse Performances // Knowledge-Based and Intelligent Information and Engineering Systems / Juan D. et al. (Eds.). – Proc. of 13th Internat. Conf., KES 2009, Santiago, Chile, September 28-30, 2009, Part I. – Lect. Notes Comput. Sci. – 2009. – Vol. 5711. – P. 261–268.

[12] Chandrasekaran B., Josephson J. R., Richard Benjamins V. Ontology of Tasks and Methods // Proc. of Workshop on Knowledge Acquisition, Modeling and Management (KAW'98), Banff, Canada, 1998.

[13] Zagorulko Yu.A., Popov I.G. A Software Environment based on an Integrated Knowledge Representation Model // Perspectives of System Informatics. – Proc. of the Andrei Ershov Second Internat. Conf. PSI'96, Novosibirsk, Russia, June 25–28, 1996. – P. 300–304.

[14] Zagorulko Yu.A., Popov I.G. Knowledge representation language based on the integration of production rules, frames and a subdefinite model // Joint Bull. of the NCC & IIS. Ser.: Comput. Sci. – 1998. – Iss. 8. – P. 81–100.

[15] GLPK (GNU Linear Programming Kit). –
http://www.gnu.org/software/glpk/glpk.html.

[16] Narin'yani A.S., Semenov A.L., Babichev A.B. et al. A New Approach to Solving Algebraic Systems by Means of Sub-Definite Models // Proc. of the 16-th IFIP Conf. on System Modelling and Optimization. Compiegne, France. July, 1993. – Lect. Notes in Control and Information Sciences. – Springer Verlag, 1994. – Vol. 197. – P. 355–364.

[17] Semenov A.L. Solving Optimization Problems with Help of the UniCalc Solver // Applications of Interval Computations. / R.B. Kearfott and V. Kreinovich (Eds). – Kluwer Academic Publishers. 1996. – P. 211–225.