

Multi-agent Data Fusion Systems: Design and Implementation Issues

Vladimir Gorodetski, Oleg Karsayev and Vladimir .Samoilov
Intelligent System Laboratory
St. Petersburg Institute for Informatics and Automation
!4th Liniya 39, St.Petersburg, 199178, Russia
{gor, ok, samovl}@mail.iias.spb.su

Abstract.

The objective of Data Fusion (DF) task is making decisions on the basis of distributed data sources accessible through Intra- or Internet. These sources contain heterogeneous data that can be represented in various structures (relational, transactional, etc.), can be of different nature (images, signals, numbers, etc), and accuracy, be measured in different scales (Boolean, categorical, real), can contain uncertainty, etc. An objective of a DF system is to combine data from many different sources to make decision, for instance, classification of an object or an object state, a situation assessment, etc. Within DF specific tasks three issues are of the most significance. The first is development of *meta-model of distributed data* sources; the second is development of *meta-model of combining decisions* produced on the basis of particular data sources and the third concerns *architectural issue*. According to our opinion, the first and the second tasks can be solved successfully only if we focus on the thorough development of the distributed application ontology playing a basic role in effective solution of many problems entailed by heterogeneity and distribution of data. In turn, the ontology-related solution significantly correlates to the architecture of the respective software. The above tasks and respective software tool considered from the design and implementation viewpoints are in the focus of the paper.

1. Introduction

From general point of view Data fusion (DF) is a task of data processing aiming at making decisions on the basis of distributed data sources specifying an entity (an object or an object state, a situation constituted by a number of autonomous interacting objects, intent of a situation management, etc.). The

distributed data used for DF can be of different physical nature (electromagnetic signals, images, sensor data, experts' information, etc.), be of different accuracy and reliability, particular data may be incomplete and uncertain and be represented in different data structures, contain missed values, etc.

According to the commonly accepted *JDL* view [3], Data and Information Fusion is a multilevel process comprising several types of tasks. *Level 0* corresponds to the *fusion of sensor signals* to produce data specifying semantically understandable and interpretable attributes of objects those are the interacting "actors" in an application under investigation. *Level 1* called "*Data fusion Level*" aims at processing the above data to make decisions with regard to classes of objects in question, or classes of these object states. The next level of data processing (*Level 2*) is normally called "*Information Fusion*" and its goal is to assess a situation constituted by the set of the above objects considered as a single whole called hereafter "a system". Information Fusion of *Level 3* corresponds to the task "*Impact assessment*" which mean, for example, adversary intent assessment produced on the basis of the situation development prediction. *Level 4* supposes calculation a feedback like planning resource usage, sensor management, etc. The upper level (*Level 5*) supposes *human activity* and *situation management*.

Till recently DF research were mainly focused on fusion data represented by digital images. Analysis of the last years publications proved that currently the area of DF interests is enlarged and involved many other applications. Besides situational awareness task, which is the focus of such research, examples of DF applications are network of sensors ([7]), monitoring of seismic activity ([22]) and some others. Let us outline two applications related to DF area.

1. *Detection of intrusions into computer network* ([2]). At present coordinated distributed attacks performed by a team of malefactors from spatially distributed hosts constitute the main threats for computer networks and information. "Traces" of an attack proves in different data perceived or generated by a computer network assurance system. For example, they are displayed in *tcpdump* generated via preprocessing of input traffic, in audit data trail, in sequences of system calls of operating system, in data resulting from monitoring of application servers, queries to databases and directories, in data specifying users' profiles, etc. Such data are generated on different hosts of computer network. The timely detection of a illegitimate user's activity is potentially feasible only in case of fusion data from different available sources. Formally, intrusion detection is a type of classification task which has to be solved on the basis of combining decisions produced on the basis of many data sources. ♦

2. *Analysis and prognosis of natural and man-made disaster development*. A lot of different kinds of potentially dangerous situations emerge in different regions

of many countries. They can emerge due to natural disasters (earthquakes, floods, etc.), man-caused emergencies (chemical, nuclear, etc.) and so on. The specific features of such phenomena are rapid development in time, spread in space, strong dependence on weather conditions, landscape, building's infrastructure and so on. To assess such a situation in order to predict its development and to prevent its undesirable or catastrophic consequences, it is necessary to use data from different sources. The sources of such data are weather data, information collected by airborne equipment (photo, TV, radar, infrared, etc), people's messages, simulation data, and so on. ♦

The formal methods and models used in data and information fusion are borrowed from very diverse areas, for instance, from statistical decision theory, artificial intelligence, data mining and knowledge discovery, machine learning, classification, image processing, uncertain and fuzzy information processing, Bayes networks, evidence theory, distributed systems, parallel processes, ontology, simulation, and so on, and so forth. The research area of data fusion is actually challenging. Nevertheless, one can notice that the main R&D data and information fusion problems are basically not of theoretical kind, although there exist a number of such tasks. Instead, they are mostly of *technological* kind. In other words, the most data and information fusion problems can be solved on the basis of the results achieved in adjoining sciences. But, in contrast, the technological aspects of the DF system design and implementation put many specific problems and need for much of new R&D efforts.

The focus of this paper is design and implementation of DF systems of level 1. Although to date several strategies for data fusion are proposed ([12]), the most popular and advantageous one is the strategy used a multi-level hierarchy of classifiers. In it the source-based classifiers make decisions on the basis of data of particular sources followed by meta-level decision making based on combining of the source-level decisions. The advantages of such a scheme are (1) decrease of the data sources information exchange; (2) simplicity of data source classifier fusion even if they use data of different representation structure, certainty, accuracy, etc.; (3) possibility to use mathematically sound mechanisms for combining decisions of multiple classifiers. In large scale applications this strategy is more accurate and computationally efficient. It should be noticed, that in some applications this strategy is the only applicable one. An example is given by the group of applications in which the data are private and the data holders do not want to share the data but agree to share decisions produced on the basis of such data sources.

The rest of the paper is organized as follows. In *section 2* the main peculiarities of data and data processing inherent for the most of DF applications are described. *Section 3* outlines and compares related works devoted to the

methods of multiple classifiers decision combining. This analysis makes it possible to justify the model and architecture of DF system. *Section 4* sketches a particular aspect of meta-model of distributed data sources that substantially determines architecture of DF system learning. *Section 5* outlines the meta-models of combining decisions produced by base-level classifiers. *Section 6* describes the developed multi-agent architecture of a Data Fusion software tool and *Section 7* outlines its design and implementation technology. *Section 8* gives short information about case studies used for debugging and evaluation of the developed technology for DF systems. *In Conclusion* the main results and future work are summarized.

2. Peculiarities of data and data processing in DF task

The key DF peculiarities are the result of the fact that data sources are *physically distributed* and *heterogeneous*. As a rule, data sources are spatially distributed or, at least, they are represented in different databases and/or located on different hosts. Heterogeneity is entailed by the diversity of possible data structures, differences in data specification languages, differences in data natures (geographical, statistical, etc.) and so on. As a rule, this data are of large scale.

Distribution and heterogeneity of data entail several new difficult problems which significantly influence on all aspects of DF system design and implementation.

The *first problem* is development of the *shared thesaurus* providing for *monosemantic understanding of the terminology* used in formal specification of domain entities. This problem arises due to the fact that specifications of data belonging to particular sources are being developed by different experts and in most cases these processes are independent. The experts can denote different domain entities by the same terms and vice versa, they can denote the same entities by different terms. Other class of problems arises due to the fact that the same entities can be represented in different sources by different data structures but in the DF procedure all of them have to be understood and used equally. According to the modern understanding, to cope with the problem in question, it is necessary to use a meta-model of data and knowledge presented in terms of *ontology* and shared by all entities of the DF system. The structure of ontology used in our research is depicted in Fig.1. It comprises *problem ontology* added with *application ontology* and also with *task ontology*. In turn, application ontology comprises two types of components: (1) a part of application ontology that is shared by all software components of DF system and (2) parts of the

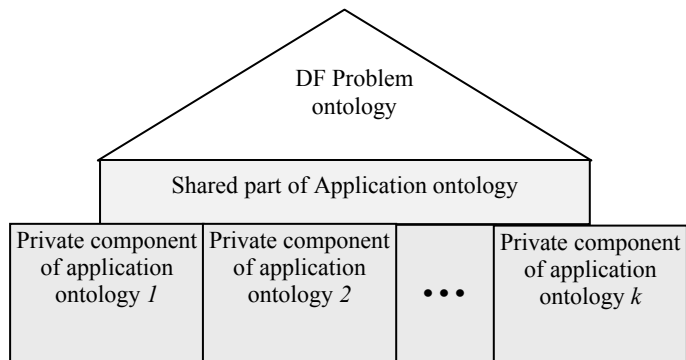


Fig.1. Tower of DF ontology components

application ontology that are private for particular data source-associated software. The private components of application ontology form an inside erminology of particular data sources¹. Our point of view on the ontology specification, representation, design and usage is close to one presented in [KAON]. In the developed architecture of DF software tool (see section 6 below) forming of application ontology is the area of responsibility of several agents participating in the iterative and interactive design of private and shared components of the application ontology.

The *second problem* corresponds to so-called *entity identification problem* ([12]). The data specifying an object is represented in different data sources. This is why each local data source only partially specifies the above object. Its complete specification is made up of data fragments distributed over the data sources. Therefore, to form a complete object specification a mechanism to identify such fragments is needed. It should be noticed that some fragments of data associated with an object can be absent in a number of sources. A graphical explanation of this problem is given in Fig.2.

Within DF software tool under development, the *entity identification problem* is solved in the following manner. In the application ontology for each entity the notion of entity identifier ("*ID entity*") is introduced. This entity identifier plays the role of the entity primary key (in analogy with the primary key of a table). For each such identifier, a rule is defined within the application ontology, which can be used to calculate the value of this key. For example, the

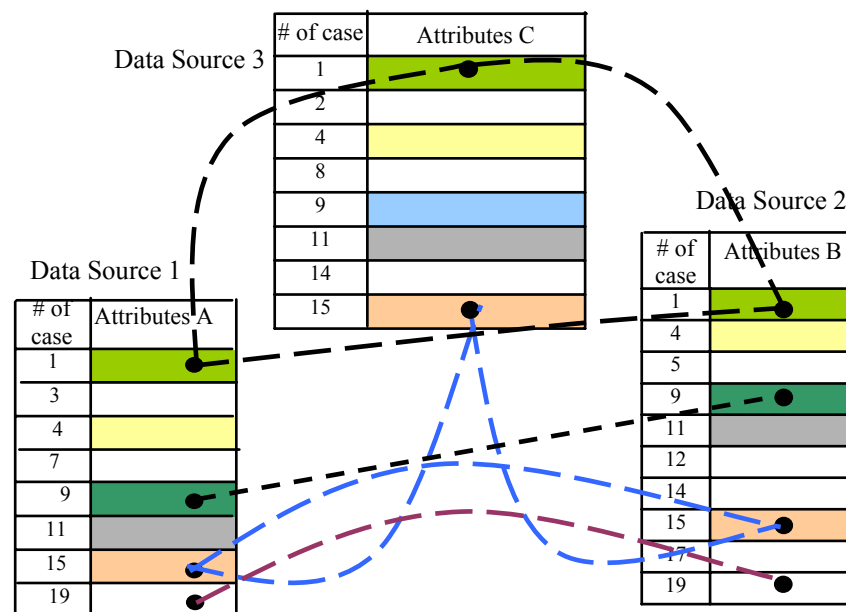


Fig.2. Illustration of the essence of the data identification problem

value of a unique combination of a subset of this entity attributes can be used as the argument of such a rule. A specific rule is defined for each particular data source that uniquely connects the entity identifier and the local primary key in this source. In a special case, it may simply be a list of pairs "*value of entity identifier*" – "*value of local key*". After such rules have been constituted for each particular source a list of all the instances of entities stored in the local sources is formed on the meta-level, and instances specified in several sources are identified.

Next, the data specifying in different sources the *same entity attribute* can be of different structures. This problem is called *non-coherency of data measurement scales*. In some cases an attribute presented in different sources may be measured in the same measurement scale but in different units. In our DF technology this problem is solved as follows. Let us designate the attribute measured differently in different sources as *X*. In the shared part of the application ontology the type and the unit of the attribute *X* measurement are determined. At the next step in all the sources where it is presented expressions are determined for this attribute through which it can further be converted into the same scale in all the sources. This allows using the values of attributes on the meta-level regardless of the data source from which they are originated. At that,

¹ Notice that development of application ontology is the subject of particular scientific research area (see, for instance [17], [18], etc.).

the selection of attribute X specification and measurement within shared part of application ontology is made by experts via negotiations according to a protocol.

The above problems together with several other ones constitute so-called *data non-congruency problem* ([12]), which exerts influence on algorithmic basis, conceptual model and architecture of DF software tool considered below.

3. Combining decisions of multiple classifiers: Related works

In our research we follow the strategy of data fusion in which DF is considered as hierarchy of multiple classifiers producing decisions on the basis of particular data sources followed by combining these decisions at the meta-level. In the most DF tasks "*decision*" is understood as classification of an entity (object, state of an object, situation, etc.), i.e. assigning the entity a class label from a fixed set. In the considered scheme of data fusion each local data source is associated with a single or several classifiers (hereafter they are referred to as *base-level classifiers* or simply as *base classifiers*). Each of them produces classification of an entity using only local data or local data fragment and then transmits the classification produced to meta-level, where these classifications are combined by meta-level classifier(s) using a method.

To the present day several basic approaches to combining decisions of multiple base-level classifiers are developed. They can be grouped into four groups:

1. Voting algorithms;
2. Probability-based or fuzzy algorithms;
3. Meta-learning algorithms based on stacked generalization idea;
4. Meta-learning algorithms based on classifiers' competence evaluation.

Voting methods were developed about twenty years ago and were historically the first ([27]). Due to their simplicity and satisfactory accuracy in many applications these methods are to date in use. The most simple of them is called "*majority voting*" ([8], [9]). The "*weighted voting*" approach ([25], [19], [4], [1], [20], etc.), which is a little more sophisticated, was implemented in many particular forms and is currently in broad use.

The methods of the second group are based on probabilistic models like Bayesian model of a posteriori probability assessment, Bayesian networks, Dempster-Shafer theory of evidence, and also on fuzzy set-based models ([3]). These models are in some sense "classical", they have long history and do not need any comments or references. Unfortunately the area of practical application of such methods in DF tasks is not very broad. In any case the question "Where probabilities come from?" has to be here answered. To our opinion probabilistic

and similar models are applicable if the dimensionality of the classification task is small and there is enough information that makes it possible to empirically assess the needed probabilistic characteristics with satisfactory accuracy and reliability. For instance, this is possible if the size of training and testing data is large.

However, at present the Knowledge Discovery from Databases (KDD) R&D community pays the most attention to the methods of combining decisions that use some kind of knowledge about properties of base-level classifiers ([29]). General idea of this group of approaches that was proposed in [34] is called "*stacked generalization*". Although the idea of "stacked generalization itself is very simple, it turned out very effective and gave birth to several particular methods of combining decisions. The most promising variations of stacked generalization were proposed in [30], [5], [26] ("*meta-classification*"), [11] ("*cascade generalization*"), and in some others. According to these approaches base classifiers, which decisions are supposed to be combined, are to be "diverse", i.e. they must be at least either based on different learning algorithms or to be learned on the basis of different training and testing data. The results of simulations proved that the former way of providing classifiers' diversity is more preferable as compared with the last one. Let us notice, that in the currently well-known and prevailing methods like *Bagging* ([6]) and *Boosting* ([10]) the last approach to providing diversity of classifiers is used. Generalized structure of decision combining based of stacked generalization (for its "meta-classification" variant) is depicted in Fig.3.

In general, stacked generalization-based methods of decisions combining are popular and still being actively researched. A drawback of this group of methods is their inability to preserve already existing set of classifiers unchanged if a new classifier inserted in the classification system ([33]). In contrast, the methods of the fourth group discussed below are free from this drawback.

These methods are based on the evaluations of classifiers' competences with regard to each particular record of input data, specifying an object. The main idea of competence-based methods of decision combining is to assign to each particular classifier the region in attribute space where it is most competent as compared with the other base-level classifiers. Firstly this idea was proposed in the papers [29] and [23]. The core of this idea is that a special procedure called "*referee*" (see Fig.4) is associated with each particular classifier. A responsibility of referee is to assess the competence of the respective classifier with regard to the particular input data ([24]). To be able to solve this task the referee has to be learnt. Referee learning is reduced to the routine learning task, which can be solved on the basis of the same training and testing data that has been used for training and testing of classifiers themselves. A specific of the last task is that in

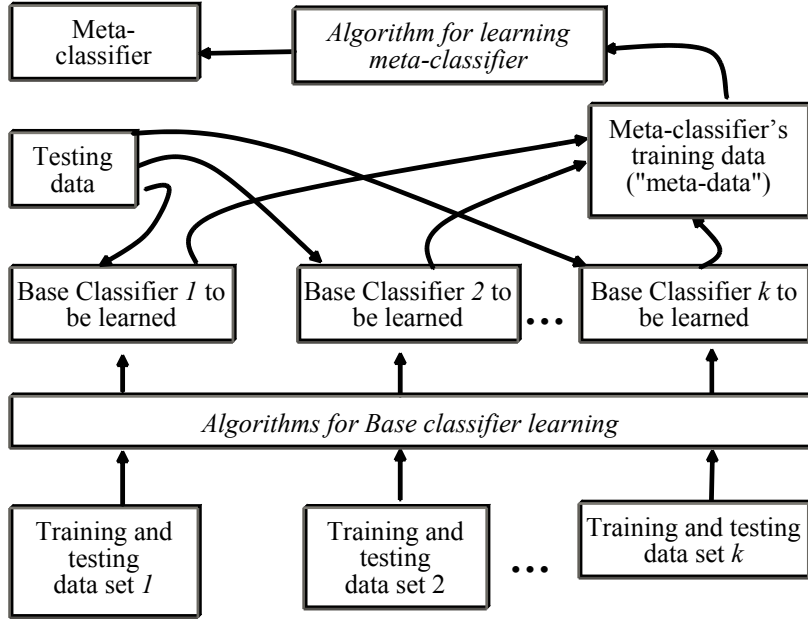


Fig.3. Meta-classification scheme

it other partition of learning data is used. In referee training task the same training and testing data are partitioned into two subsets of positive and negative examples like this is shown in Fig.4. In competence-based methods decision combining consists of two steps: (1) detection of the most competent classifier and (2) selection of the classification produced by the most competent one.

Further important development of this method was proposed in the papers [28], [31] and [32], at that the last two papers proposed the definitely significant improvement of the method.

In general, the competence-based approach is very promising and simulation proved this fact ([23], [24], [29], [32]). Its advantages are higher accuracy (as compared with the voting and stacked generalization-based methods) and also its capability to preserve already existing set of classifiers unchanged if a new classifier inserted in the multi-level classification system.

Two types of methods discussed in this section, i.e. meta-classification and competence-based methods are included into the multi-agent data fusion learning component of the DF software tool under development. It is noteworthy

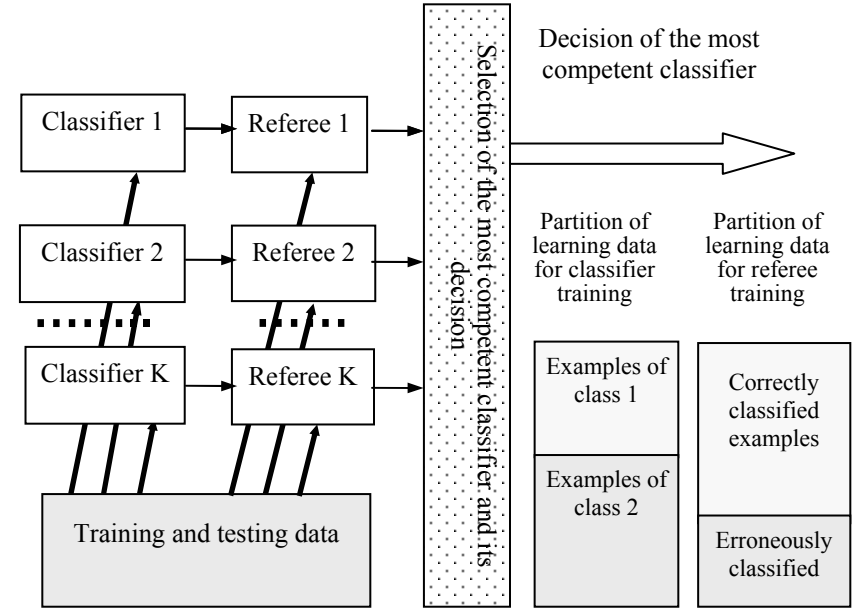


Fig.4. Explanation of the idea of competence-based approach to combining decisions of multiple classifiers

to mention that in DF applications both above methods cannot be used in "straightforward" manner because of peculiarities of data and needs to be adapted (This question with regard to DF task was discussed above.) and the necessity to use distributed learning and decision making procedures. Again, most of these peculiarities are of technological kind. Some aspects of this question are discussed in next section.

4. Meta-model of Training and Testing Data

Data inherent to DF tasks possess several peculiarities, which lead to the necessity to revisit or modify the existing approaches to decision combining learning. Let us recall some of these peculiarities important from learning viewpoint::

- Data are distributed in space and stored in different databases;
- Each data source only partially specifies the same object to be classified in terms of attributes which can be different in different data sources;

- Data can be incomplete; it can contain particular attribute values and also the total records in a source missed (see Fig.2, Fig.5).

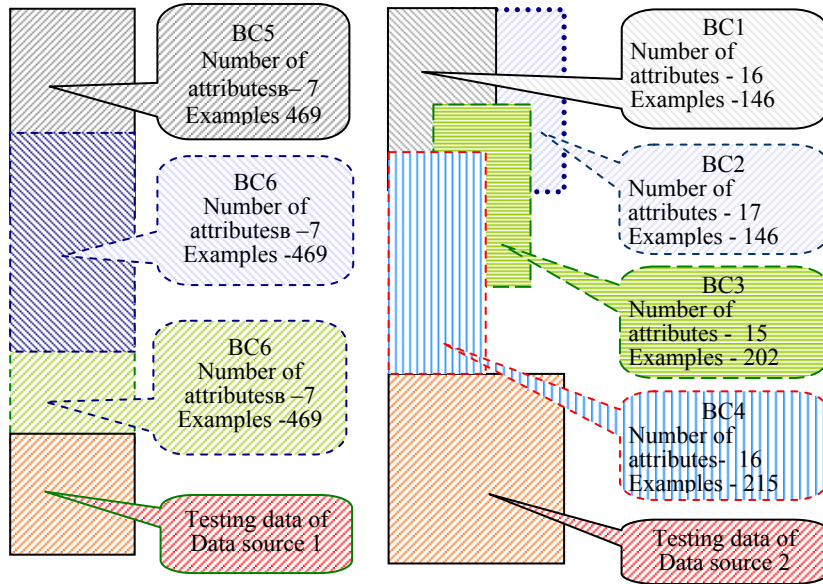


Fig.5 a. Data source 1

Fig.5 b. Data source 2

Fig.5. Example of data partitions in different data sources

Hence, in case of distributed learning inherent for DF it is necessary to take into account potential distinctions in subsets of attributes specifying an object in different data sources. This fact is explained in Fig.5 (a, b). In these figures, two data source examples are depicted schematically¹. The figures reflect the facts that data used for training and testing of different classifiers can be overlapping in two aspects: the subsets of attributes of different sources can be overlapping and the subsets of data records used as training and testing data can also be overlapping. Let us notice that data sources presented in Fig.5 (a, b) were created artificially from the data set [35] via splitting the latter in a way that makes it possible to model properties of real-life data sources. The data of both DS1 and DS2 data sources have also been split into subsets of training data

¹. They correspond to the data set that was proposed in KDDCup-99 competition of data mining and knowledge discovery software [35]. This data set is used in case study to be outlined in section 8.

according to the chosen structure of source-based classifiers. A particular subsets of records, as it is shown in Fig.5 (a, b), are preserved for use as training and testing data at meta-level (for meta-classifier or for referee depending on knowledge engineer choice).

In DS1 (Fig.5a) all three base-level classifiers use the same set of attributes but different (non-overlapping) subsets of training and testing data records. These classifiers can also be trained on the basis of different learning procedures. For data sources split like DS1 both meta-classification and competence-based approaches are applicable.

The partition of data of the second data source demonstrates both more general and more complicate case. In it, the base-level-classifiers can differ along three "dimensions": they can use different training and testing data, they are based on different subsets of data attributes and they can be learned on the basis of different learning algorithms. At this data splitting both scheme of decision combining can often be used but competence-based approach is applicable only if there are no missed records in all data sources and data identification problem is resolved successfully. Otherwise, for some examples of testing and training data the competence of base-level classifier can not be evaluated.

5. Structures of decision combining

DF software tool has to contain components that make it possible development of two interacting components. The first of them is component

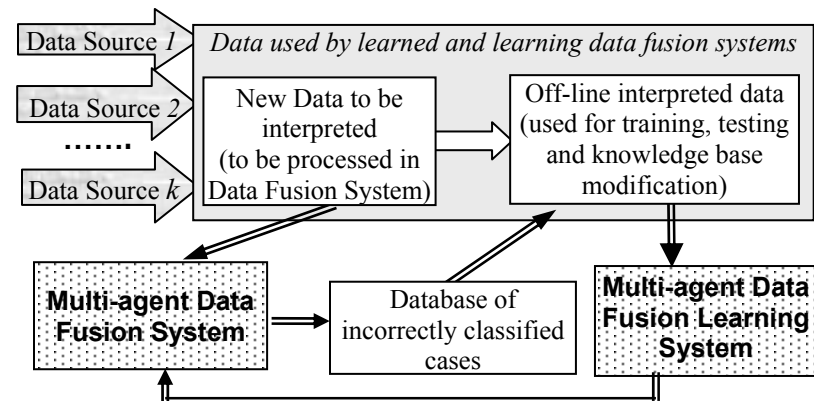


Fig.6. Interaction of Data Fusion and Data Fusion Learning components aiming at development of DF system itself and the second one is the component

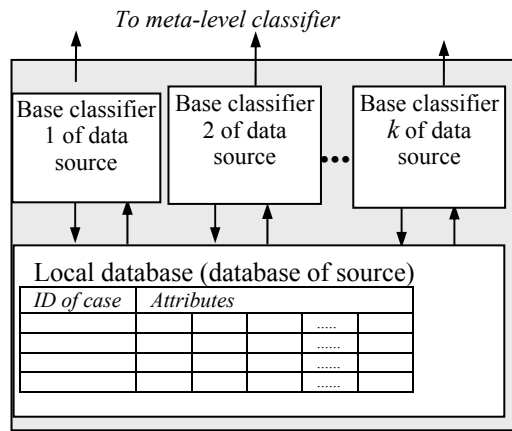


Fig.7. Combining base-level decisions: Variant 1

explained in Fig.6.

Formal frameworks for combining decision were discussed in previous section. Let us consider the variants of structures according to whose the base-level and meta-level classifiers interact during solving a DF task. The choice of

aiming at learning DF system. We suppose that both of these components interact during the whole life cycle of DF system, i.e. during development of DF system (at this phase the second component is responsible for DF base-level and meta-level classifier learning) and during its maintenance where the second component is responsible for incremental learning of the first one. The scheme of interaction of the above components is

a structure influences in a degree on the architecture of DF component, on DF learning component and on interaction of agents within the architecture in question. In Fig.7 the case where decisions of only base-level classifiers are sent to meta-level that carries out their decision combining. Fig.8 demonstrates the case where meta-level classification is used even on local source level. In this case the results of source data-based meta-level classification are forwarded to meta-level.

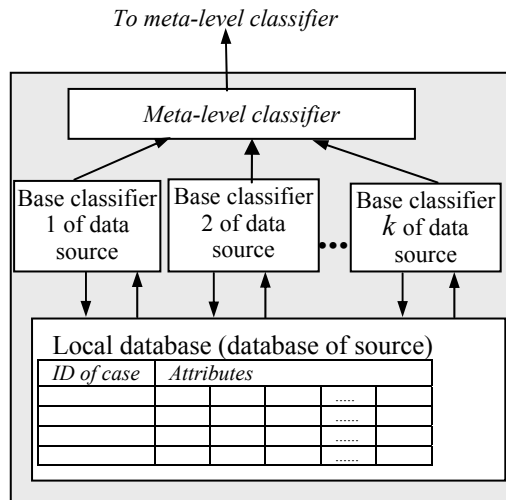


Fig.8. Combining base-level decisions: Variant 2

both meta-level decision and part of base-level decisions are forwarded to meta-

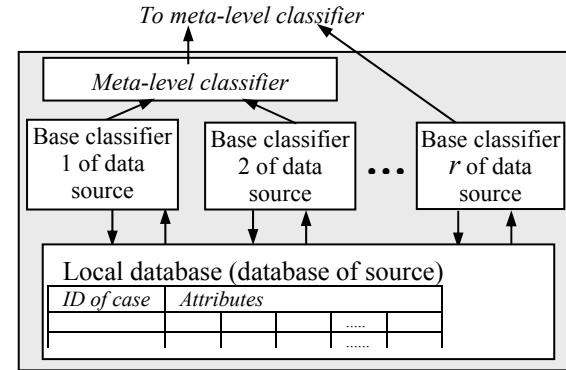


Fig.9. Combining base-level decisions: Variant 3

level in order to be combined together with the decisions produced by classifiers of the other local data sources. Let us notice that in particular cases several meta-level classifiers can be required in data sources (as well as at meta-level) if learning data are of too large size and dimensionality.

6. Multi-agent architecture of data fusion and data fusion learning components

Any DF system is distributed in its nature. Actually it deals with distributed data sources and realizes distributed or decentralized data processing. On the other hand, design and implementation technology of a DF system is distributed too and also it is iterative and interactive process involving a lot of human efforts. According to our opinion multi-agent architecture is the most appropriate solution for both DF software tool and desired DF system. The developed architecture of DF software tool consists of two kinds of components:

- (1) Components responsible for the design of source based parts of the desired DF system and
- (2) Component supporting iterative and interactive design of the meta-level part of the desired DF system.

The architecture components are presented in Fig.10 and 11. Both of them present allocation of particular tasks over agents and other components. Each data source-based component comprises the following parts (Fig.10):

1. *Data source managing agent*. It is responsible for design of the private component of the application ontology and its consistency maintenance with regard to shared component of ontology. In addition, it plays the role of the gateway to the source database.
2. *KDD agent*. Its responsibilities are training and testing of all source-based classification agents of the DF system under development including meta-classifier(s) and/or referee(s) if any.

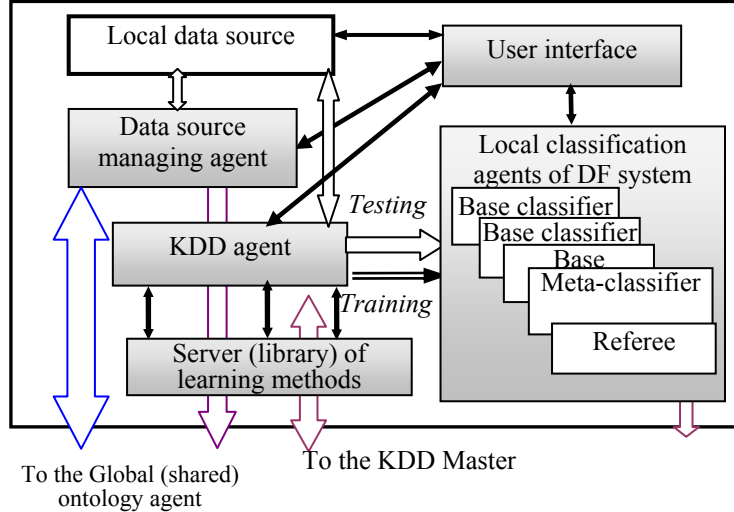


Fig.10. Architecture of the source-based component of DF software tool

3. *Local classification agents of DF system*. These agents constitute a part of the desired DF system producing source-based decisions, i.e. they are learned classification agents operating with the source data.

4. *Server of learning method*. It comprises a multitude of KDD methods, metrics for evaluation of the learning quality and other functionalities needed to learn classification agents of sources.

This component of the architecture also includes local database and user interface providing interactive mode of its operation.

The meta-level component of the architecture comprises the following agents (Fig.11):

1. *Meta-Learning agent ("KDD Master")*. Its functions are as follows:
 - Management of the design and consistency maintenance of DF system application ontology,
 - Design of meta-model of training and testing data.
 - Design of meta-model of decision making.
2. *Meta-level KDD agent*. It aims at solving the tasks of training and testing of meta-level classification agents (meta-classifiers or /and referees);
3. *Agent-classifier of meta-level*. This agent is part of the DF system. It stores meta-level knowledge base created by *Meta-level KDD agent*, receives decisions from local source-based agents of DF system and

produces top-level decision, i.e. actually it is meta-level classification agent to be trained;

4. *Server (library) of KDD methods*. It stores KDD methods, metrics for evaluation of the learning quality and other functionalities needed for operation of *Meta-level KDD agent*.

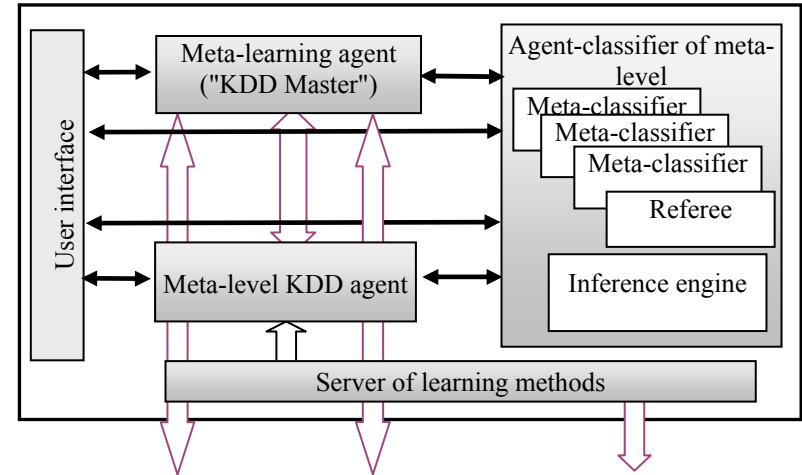


Fig.11. Architecture of meta-level component of DF software tool

Each particular agent is built on the basis of so-called "*Generic Agent*" (see for more detail next section) and all agents have the same generic architectures (Fig.12). The distinctions among the agents are reflected in the content of particular agents' databases and state machine's scripts specifying the agent behavior and knowledge bases. Let us outline agents' generic architecture.

Each agent's neighborhood comprises other agents, with whom it communicates, environment, which it perceives, and user interacting with agents through user interface. *Receiver of input messages* and *Sender of output messages* play the roles of mediators between the agent and environment. Messages received are recorded in *Input message buffer*. The order of its processing is managed by *Input message processor*. In addition, this component performs syntax analysis and interpretation of KQML messages, and also extracts the message contents represented in XML language. The component *Database of agent's dialogs* stores attributes of each input message: identifiers, type of message and its source. *Meta-state machine* manages the semantic processing of input messages and forwarding it for processing to the respective

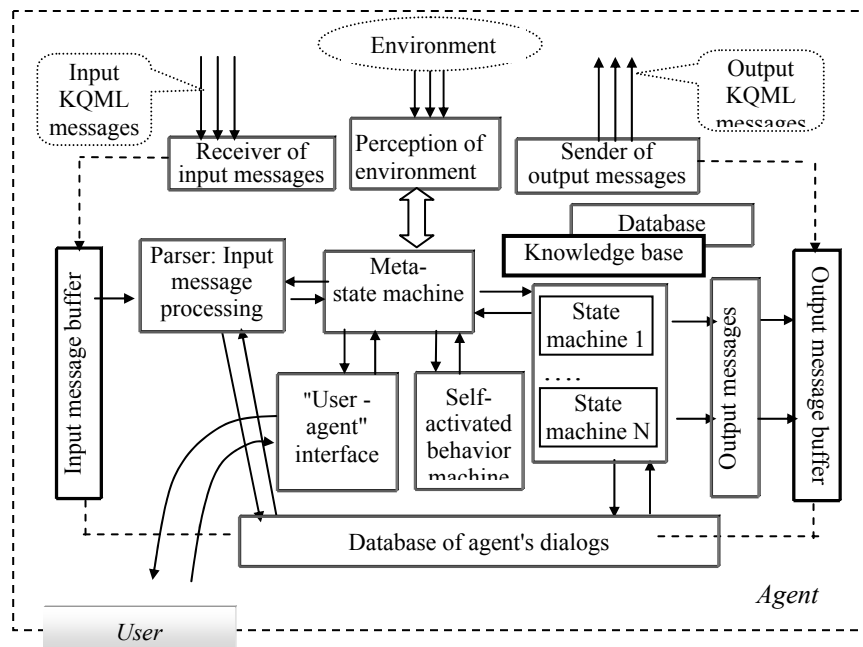


Fig.12. Generic architecture of an agent

State machine(s). An additional functionality of this component is management of parallel performance of agent's processes. The basic computations of agent corresponding its role in MAS are executed by *State machine 1, ..., State machine N*. Each of them is implemented as a component realizing a scenario of processing of definite kinds of input messages.

The selection of scenario and therefore the output result depends on the input message content and inner state of the *State Machine*. In turn, inner state of the latter depends on the pre-history of its operation; in particular, this pre-history is reflected in the state of agent's *Knowledge-* and *Database*. One more state machine called "*Self-activated behavior machine*" is changing its inner state depending on the state of the data- and knowledge bases. In some combinations of these states it can activate functionality without an external command issued by input messages or environment.

7. Summary of implementation technology

Multi-agent DF system is being developed by use of *Multi-Agent System Development Kit (MAS DK)* that was developed by authors [13]. It provides

support for basic phases of multi-agent application design and implementation. Let us outline the main components of this tool and peculiarities of the technology supported by it.

MAS DK consists of two main components. The first one is a so-called "*Generic Agent*" that comprises the library of reusable classes and standard database structures. Its second component is a set of specialized editors supporting specialization of *Generic agent* according to the application architecture and functionalities of particular agent composing applied MAS. The development technology supported by MAS DK comprises two phases. At the first one the application is specified in terms of MAS DK specification language and reusable classes of *Generic agent*. This phase results in so called "*System Kernel*". At the next phase *System Kernel* is used for MAS deployment. At the deployment phase the software code of applied MAS is generated and software agents are situated in computers of the network according to the MAS specification in *System Kernel*.

Generic Agent typifies classes of agents to be specified in *System Kernel* and is used further for replication of the software agent instances. *Generic Agent* consists of three components: (1) Invariant scheme of agents' databases, (2) Library of invariant Visual C++ and Java classes implementing the basic mechanisms of data processing and (3) Library of reusable Visual C++ and Java classes supporting agent communication via message exchange. The latter includes implementation of KQML language, message content specification language based on XML, message templates, and parsers needed to extract and interpret message content.

Generic Agent data structures are used to represent the data- and knowledge bases of agent classes. Data- and knowledge bases of an agent class are specified in terms of the domain ontology intending to provide their integrity and agents' mutual understanding via using structured common terminology and its shared interpretation. Also, ontology defines functionalities of the particular agent classes. The above specification form *System Kernel*. Apart from specification of agent classes generated from *Generic agent*, *System Kernel* also contains information about the numbers of instances of each agent class to be replicated and about the names of hosts of the local network, in which they are to be situated.

MAS DK consists of several specialized editors as follows:

1. *Ontology editor* is a user's tool that is utilized for specification the structured domain ontology concepts and their attributes.
2. *Editor of agent classes* support specification of the list of the agent classes to be generated according to the model of the applied MAS under development. Every such an agent class inherits the methods and data structures of *Generic*

agent. In the DF applications the examples of agent classes are presented in Fig.10 and Fig.11. At posterior development steps each class of agents can be represented in MAS by several instances situated in different computers.

3. *Agent class ontology editor* aims to support allocation of the subset of domain ontology concepts, which the particular agent deals with, and generation of agent class database according to the above allocation. If necessary, this editor supports definition and specification of new concepts of the agent class. The agent class database structure is generated on the basis of the entire set of its concepts.
4. *Editor of agents' behavior scenarios* supports specification of the scenarios of input message processing and rules of knowledge base determining the scenario to be executed.
5. *Editor of message templates* supports specification of the set of templates of output messages of each agent class.
6. *Agents' replication manager* aims at support generation of the instances of each agent class according to the MAS conceptual model and architecture. For example, in DF applications this procedure performs replication of local data source-based agents and tunes their parameters, e.g. agent's identifier, the name of host to situate, etc.

The second step of technology is installation of the software agents within the computer network (according to locations of data sources—in case of DF applications). This procedure is carried out in a semi-automatic mode. At that, the information that is to be defined by the developer is to point out the names of hosts (*IP-addresses*), in which each agent is intended to be situated as it is specified in *System Kernel*. Special software generates the parameters of the message templates to tune addressee(s) for each message, and copies the data associated with the agent to a folder in the host computer. Several simple operations follow up the last ones.

8. Applications

Three multi-agent DF applied system prototypes are currently the subjects of design and implementation. They are outlined below.

KDDCup-99 –based Case study of DF system

1. Application corresponding to the KDDCup-99 data set [35] deals with Intrusion Detection Learning Task. Inherently this data are not distributed but they were split artificially to model multiple sources and then to use them as a case study. The KDDCup-99 data set is specified by 36 attributes of different types (numerical–28, categorical–4, Boolean–4), and the total size of training

and testing data records is equal to 33460 at that $TT=7100$ of them were used for training and testing base-level classifiers and meta-classifier and the rest, i.e. $FT=26360$, were used for final evaluation of the accuracy and performance of the developed multi-agent DF system. The TT data set was artificially split into two data sources DS1 and DS2 (they have 1 common Boolean attribute). In turn, the DS1 and DS2 data sets were also split like it is shown graphically in Fig.5a and Fig.5.b. The last splitting aims to form training and testing data for particular base-level classifiers and meta-classifier at that the total number of base-level classifiers was chosen equal to 7 (3 of them were used in DS1 and 4–in DS2). The scheme of interaction of base-level classifiers and meta-level classifier is demonstrated in Fig.13. The base-level classifiers differ in attribute sets and also in training and testing data sets and some of them also differ in learning algorithms used. Two basic algorithms of learning were used in this case study: "*Visual analytical Mining*" ([14], [15]) intended to mine numerical data, and GK2 algorithm ([16]) intended to mine discrete data.

A description of data sources splitting and also accuracy of base-level classifiers and meta-classifier on training and testing datasets and also on total batch of KDDCup data sets are presented in Tab.1.

Table 1. Properties of data sources and accuracy for KDDCup case study

	Attribute's measurement scale	Sizes of training and testing data.	Accuracy on training data	Accuracy on testing data.	Number of rules.	Accuracy on the <i>FT</i> data set
BC1	Boolean – 1 Numerical – 7	146/178	0.83	0.87	4	0.02
BC2	Boolean – 3 Numerical – 7	146/178	0.82	0.84	4	0.99
BC3	Boolean – 1 Numerical – 5	202/178	0.85	0.85	3	0.02
BC4	Boolean – 1 Numerical – 4	215/178	0.85	0.88	4	0.02
BC5	Boolean – 1 Numerical – 2	469/1739	1.0	1.0	2	0.99
BC6	Boolean – 1 Numerical – 2	469/1739	1.0/	1.0	2	1,0
BC7	Boolean – 0 Numerical – 2	469/1739	1.0	1.0	1	0.99
MC	Boolean – 7 Numerical – 0	100/400	0.95	0.97	6	0.99

This case study was intended to be used for evaluation of correctness, advantages and drawbacks of the partially developed DF software tool and also multi-agent approach to the design and implementation of DF systems and justify its feasibility as well as quality of the resulting DF system operation.

Other applications in progress

At present two other applied multi-agent data fusion systems are the subjects of the developments.

The first of them is Multi-agent intrusion detection system and its learning component (see [2] and also see the summary of this application in *section 1*).

The second case study which objective is image classification is currently also being developed. It uses GRSS DFC data set of UCI repository [36]. Both of this developments are at the beginning stage but due to effectiveness of MAS DK that is being used as software tool both of them will hopefully be finalized in a couple of months.

9. Conclusion

The paper analyzes technological issues of the design and implementation of Multi-agent Data Fusion applications. According to its nature, this task supposes a distributed implementation. Although at prima facie it could seem that the most difficult task within this problem is distributed knowledge discovery and distributed decisions making, this problem put several new non-specific tasks and challenges within both technology of data mining and knowledge discovery area and in the area of design and implementation of distributed applications.

The key DF peculiarity results from the fact that data sources are physically distributed and heterogeneous. As a rule, data sources are spatially distributed or, at least, they are represented in different databases and/or located on different hosts. Heterogeneity is entailed by the diversity of possible data structures, differences in data specification languages, differences in data natures (geographical, statistical, etc.), etc. These and some other specific properties of data to be processed in DF applications constitute several new challenging problems.

These problems are in the focus of the paper. It presents meta-model of data sources and technology for its design, meta-model of distributed learning and distributed decision making and also a technology for their design. It is accepted that all these technologies are supported by Multi-agent Data Fusion software tool which architecture is developed and described in the paper. Within this architecture the primary role belongs to the specialized agents responsible for interactive and iterative design of consistent application ontology and agents

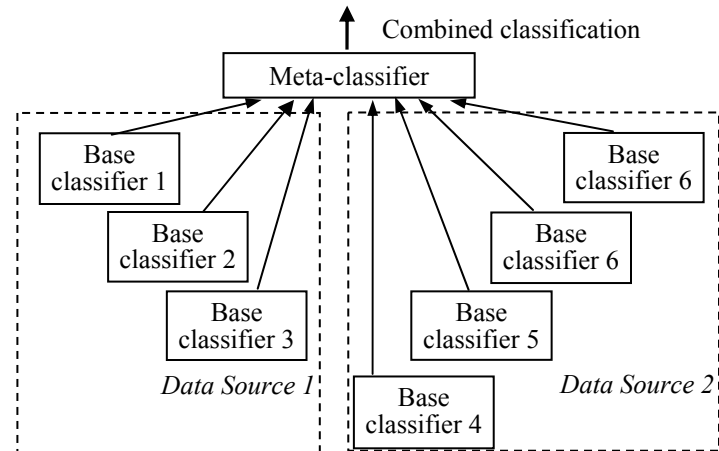


Fig.13. Classification structure in KDDCup case study

providing design of decision making structure and multi-level decision making learning. Although implementation of this tool is now in progress, it already makes it possible to develop applications in semi-automated mode. In particular, it has already been used in development a case study outlined in *section 8*. It is also being used for the development of two other DF applications.

The paper also briefly presents Multi-Agent System Development Kit (MAS DK) of general purposes, which was developed by authors of the paper and used for the development of DF software tool.

Future research will be focused on the further development of technological aspects of Data Fusion software tool and a number of particular applications in this area to accumulate experience and make it real-life task oriented.

Acknowledgements

This research is supported by grant of AFRL/IF–European Office of Aerospace Research and Development (Project 1993P) and also by grant # 01-01-00109 of the Russian Foundation of Basic Research. Participation in *ICTSM'10* is funded by the Office of Naval Research International Field Office.

References

- [1] K.Ali and M.Pazzani. "Error reduction through learning multiple descriptions". *Machine Learning*, 24(3), 173-202, 1996.

- [2] T.Bass. "Intrusion Detection System and Multisensor Data Fusion: Creating Cyberspace Situational Awareness". *Communication of the ACM*, vol.43, No. 4, 99-105 (2000).
- [3] E.Blash. "Tutorial TD2: Fundamentals of Information Fusion and Applications". *Information Fusion -2002*, Annapolis, MD, USA, July 2002.
- [4] W.L.Buntine. "A theory of learning classification rules". *Ph.D thesis*, University of Technology, School of Computing Science, Sydney, Australia, 1990.
- [5] P.K.Chan and S.J.Stolfo. "A comparative evaluation of voting and meta-learning on partitioned data". In *Proceedings of Twelfth 4th International Conference on machine Learning*, Tahoe City, CA, 90-98, 1995.
- [6] R.T.Clemen. "Combining forecasts: A review and annotated bibliography". *International Journal of Forecasting*, 5, 559-583, 1989.
- [7] D.Corkill and V.Lesser. "The use of meta-level control for coordination in distributed problem solving network". In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83)*, Menlo Park, CA, 767-770.
- [8] T.Dietterich. "Machine Learning Research: Four Current Directions". *AI magazine*. 18(4), 97-136, 1997.
- [9] T.Dietterich. "Ensemble Methods in Machine Learning". In *M.Arbib (Ed.) Handbook of Brain Theory and Neural Networks*, 2nd Edition, MIT Press, 2001.
- [10] Y.Freund and R.Shapire. "Experiments with a new boosting algorithm". In L.Saitta (Ed.) *Machine Learning. Proceedings of the 13th International Conference*. Morgan Kaufmann, 1996.
- [11] J.Gama and P.Brazdil. "Cascade generalization". *Machine Learning*, 41(3), 315-342, 2000.
- [12] I.Goodman, R.Mahler, and H.Nguen." *Mathematics of Data Fusion*". Kluwer Academic Publishers, 1997.
- [13] V.Gorodetski, O.Karsayev, I.Kotenko, and A.Khabalov. "Software Development Kit for Multi-agent Systems Design and Implementation". In B.Dunin-Keplicz and E.Nawareski (Eds.) "From Theory to Practice in Multi-agent Systems". *Lecture Notes In Artificial Intelligence*, vol. 2296, 121-130, Springer Verlag, 2002.
- [14] V.Gorodetski, V.Skormin, and L.Popyack. "Data Mining Technology for Failure Prognostics of Avionics", *IEEE Transactions on Aerospace and Electronic Systems*, accepted for publication in 2002.
- [15] V.Gorodetski, V.Skormin, L.Popyack, and O.Karsayev". Distributed Learning in a Data Fusion System". In *Proceedings of the Conference of the World Computer Congress (WCC-2000) "Intelligent Information Processing" (IIP2000)*, Beijing, 147-154, 2000.
- [16] V.Gorodetski and O.Karsayev. "Algorithm of Rule Extraction from Learning Data". In *Proceedings of the 8th International Conference "Expert Systems & Artificial Intelligence" (EXPERTSYS-96)*, 133-138, 1996.
- [17] T.Gruber. "Towards Principles for the Design of Ontologies Used for Knowledge Sharing". *International Journal of Human-Computer Studies*, 43, No. 5/6, 907-928, 1994.
- [18] N.Guarino. "Formal Ontology and Information Systems". In N.Guarino (Ed.) *Proceedings of International Conference on Formal Ontology in Information Systems (FOIS'98)*. Amsterdam, IOS Press, 3-15, 1998.
- [19] S.Hashem. "Optimal linear combination of neural networks". *Ph.D. thesis*, Purdue University, School of Industrial Engineering, Lafayette, IN, 1997.
- [20] M.Jordan and R.Jacobs. "Hierarchical mixtures of experts and the EM algorithm". *Neural Computations*, 6(2), 181-214, 1994.
- [21] "KAON-The KARlsruhe Ontology and Semantic Web Infrastructure". <http://kaon.semanticweb.org/>
- [22] C.Mason and K.Johnson. "DATMS: A framework for distributed assumption-based reasoning". In *Distributed Artificial Intelligence*, vol. 2, Eds.M.Hurbis and L.Gasser. CA, Morgan Kaufmann, 1999, 293-318.
- [23] C.J.Merz. "Combining classifiers using correspondence analysis". In *Advances in Neural Information Processing*, 1997.
- [24] J.Ortega, M.Coppel, and S.Argamon. "Arbitrating Among Competing Classifiers Using Learned Referees". *Knowledge and Information Systems* 3 (2001) 4, 470-490, 2001.
- [25] M.Perrone and L.Cooper. "When networks disagree: Ensemble methods for hybrid neural networks". In R.J.Mammone (Ed.), **Neural Networks for Speech and Image Processing**, Chapman and Hall, 1993.
- [26] A. Prodomidis, P. Chan, and S. Stolfo. "Meta-learning in distributed data mining systems: Issues and approaches". In P.Chan and H.Kargupta (Eds.) **Advances in Distributed Data Mining**, AAAI Press, 1999. Also available at <http://www.cs.columbia.edu/~sal/hpapers/DDMBOOK.ps.gz>.
- [27] L.Rastrigin and R.Erenstein. "Methods of collective pattern recognition". Moscow, Energoizdat Publishers, 1981. (In Russian.)
- [28] A.K.Seewald and J.Fuernkranz. "An evaluation of grading classifiers". In *Proceedings of 4th International Conference "Intelligent data Analysis"*, LNCS 2189, 115-124, 2001.
- [29] K.Ting. "The characterization of predictive accuracy and decision combination". In *Proceedings of 13th International Conference on Machine Learning*, Morgan Kaufman, 498-506, 1996.

- [30] K.Ting and I.Witten. "Issues in stacked generalization". *Journal of Artificial Intelligence Research*, 10, 271-289, 1999.
- [31] L.Todorovski and S.Dzeroski. "Combining classifiers with meta decision trees". In D.A.Zighen, J.Komarowski and J.Zitkov (Eds.) *Proceedings of 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD-2000)*, Lyon, France, 54-64, Springer Verlag, 2000.
- [32] L.Todorovski and S.Dzeroski. "Combining multiple models classifiers with meta decision trees". In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*, 54-64. Berlin, Springer Verlag, 2000.
- [33] R.Vilalta and Y.Drissi. "A perspective view and survey of meta-learning". Submitted to the *Journal of Artificial Intelligence Review*, <http://www.research.ibm.com/people/v/vilalta/papers/jaireview01.ps>
- [34] D.Wolpert. "Stacked generalization". *Neural Network*, 5(2), 241-260, 1992.
- [35] <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.
- [36] [http://ics.uci.edu: pub/machine-learning-databases](http://ics.uci.edu:pub/machine-learning-databases).

Vladimir I. Gorodetski, Prof. of Computer Science, Head of Intelligent Systems Laboratory of the St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, received MS degree in mechanics from the Military Air Force Engineering Academy, St. Petersburg (1960) and MS degree in mathematics from the St. Petersburg State University (1970), obtained his Ph.D. degree in 1967 and Doctor of Technical Sciences degree in 1973 in Space Vehicle Optimal Control (both from the Military Air Force Engineering Academy, St. Petersburg), Professor of St. Petersburg Institute for Informatics and Automation (1989). Member of Russian and European Association of Artificial Intelligence and member of IEEE Computer Society.

Main publications (more than 200) relate to the following areas: Optimal Control System Theory, Space Mechanics, Applied Statistics, Applied Algebra, Formal Grammars, Simulation, Planning and Scheduling, Pattern Recognition, Artificial Intelligence, Knowledge Discovery from Databases, Data Fusion, Multi-Agent Systems, Digital Image Steganography, and Computer Network Security.

Oleg V. Karsaev, Senior Researcher of Intelligent Systems Laboratory of the St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, received MC degree in Artificial Intelligence in 1981 and Ph.D. degree in Computer Science in 1986 (both from the Military Space Engineering Academy, St. Petersburg). Main publications relate to the following areas: Artificial Intelligence, Knowledge Discovery from Databases, Computer Network Security., Operation Planning and Scheduling, Multi-agent System Technology and Applications, Data Fusion, Ontology.

Vladimir V. Samoilov, Senior researcher of the Intelligent Systems Laboratory of the St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences; received MC degree in Electrical Engineering (1993) from Naval Engineering University and BC degree in Mathematics (1996) from St. Petersburg State University. Author of 11 publications in the following areas: Data Mining and Knowledge Discovery, Data Fusion, Multi-agent Systems, Databases, Object-oriented Design, Digital Image Steganography. Programming skills: Visual C++, SQL, XML, HTML, RDF, etc.