Component Interaction in Distributed Knowledge-Based Systems

Milko Marinov, Irena Valova

University of Ruse, Dept. of Computer Systems & Technologies, Ruse, Bulgaria

Abstract **– The process of group decision making requires a set of databases and knowledge bases with coordinated communication between them. A distributed knowledge-based system consists of several knowledge-based systems, and there is a possibility for their physical or logical separation. The article presents the conceptual model of a distributed knowledge-based system. The suggested architecture supports sharing and integration of knowledge and data in addition to allowing heterogeneous knowledge-based systems to communicate with each other. The article also outlines the implementation of interaction protocols comprising intelligent objects which cooperate with each other in order to find solutions to assigned tasks.**

Keywords **– Decision support systems, Distributed knowledge-based system, Interaction protocols, Multi-agent systems.**

1. Introduction

Nowadays, in modern organizations, not all use cases intuitively lead to the use of relational database management systems (DBMS) that, in turn, do not also need the strict ACID properties (particularly Consistency and Isolation). During the 80s and 90s, most of the data were stored in organizations' DBs. They have to be structured, generated and accessible in a controllable manner, and represented as "records" of the business transactions. Undoubtedly, this type of data still exists. What is more, it will continue to exist and has to be modelled and stored, so there is guaranteed access to it through DBMSs. But what happens to the vast amount of uncontrolled, unstructured, information-oriented "blast" of data that has emerged in organizations over the last 15 years with the appearance of the Web, e-commerce, social networks, etc.? Actually, organizations do not need relational DBMSs to store and restore data because their basic features are not suitable for the nature, characteristics and use of this data [4], [5]. With reference to this, different types of data models are required for the specific uses. NoSQL databases (DB) connect with the various types of non – relational data storage. They use other data models. NoSQL DBs have become an important part of the DB domain and, when used appropriately, they can offer real advantages [11].

A distributed knowledge-based system consists of several knowledge-based systems (KBS), and there is possibility for their physical or logical separation. The separate KBSs are connected and managed by the distributed knowledge-based management system (DKBMS) that coordinates the joint interaction, aimed at executing local goals, and resolves conflicts which appear between the autonomous KBSs. Knowledge distribution leads to improving the database (DB), as well as the knowledge base (KB) [6], [8]. Knowledge secures the management and general information, the latter is having complex structure and small size. At the same time, data represents the actual processed information which has conventional structure but considerable size [3], [7].

The architectural design of a knowledge-based information system requires the integration of four complex areas, i.e. distributed DBs, distributed problem-solving, object-oriented programming, and decision making [12]. Many of the issues, discussed in the literature on distributed DB, refer to the problems of the distributed knowledge-based systems, namely: globally defined DB schemas, communication between local DBs, heterogeneity, fragmentation, and localization [5], [9]. Researchers have offered several distributed DB architectures [9],

DOI: 10.18421/TEM83-04
https://dx.doi.org/10.18421/TEM83-04

Corresponding author: Milko Marinov,
University of Ruse, Dept. of Computer Systems & Technologies, Ruse, Bulgaria
Email: *MMarinov@ecs.uni-ruse.bg*

Received: 18 June 2019.
Revised: 24 July 2019.
Accepted: 03 August 2019.
Published: 28 August 2019.

[11], [14], the three-layer one being the best accepted one. The tree-layer architecture, described by Özsu & Valduriez [9], defines a conceptual framework involving relationships and properties for the global network of distributed, dependent DBs. The second and third levels of the architecture determine the fragmentation and localization schemes which distribute the conceptual scheme to the different network nodes. The object-oriented paradigm is suitable for organizing the global DB [12], [13]. This approach combines the advantages of the popular ways of knowledge representation such as logical representation, semantic networks, frames, and production rules.

The efforts in distributed artificial intelligence are aimed to understanding and modelling actions, and knowledge in multi-agent systems. The term "agent" in artificial intelligence denotes an object which functions continuously and autonomously in the environment in which other processes take place and other agents exist. Since the control and knowledge are essential for the achievement of a local or global goal distribution the solution of the problems requires cooperation and communications among the agents. The study presented in [2] discusses the main characteristics of various agent interaction protocols and focuses on the challenges each protocol offers to the research community. Among the major problems faced by the designers and developers of a multi-agent system is the following [1], [2]: How to provide the interaction and the communications between the agents, what languages or communication protocols are to be used, what and when to communicate.

The purpose of the present study is to offer a flexible distributed knowledge-based system architecture which facilitates the development of information systems facilitation the decision making process of business organizations. This process of group decision making requires a set of databases and knowledge bases with coordinated communication between them. The integrated information system productivity can be significantly increased by allowing parallel processing of separate sub-tasks, furthermore, the help of individual knowledge-based systems is necessary, too. When it is necessary to make joint decisions, every knowledge-based system should have access to multiple external data and knowledge sources.

The article is structured as following: Section 2 presents a three-layer distributed system architecture; the generalized implementation of the system is considered in Section3; Section 4 discusses the implementation of interaction protocols between intelligent objects that cooperate with each other to solve set problems; and finally, in Section 5 the authors make a conclusion and offer ideas for further research.

2. Architecture of a distributed knowledge-based system

A distributed knowledge-based system consists of one or several knowledge-based systems which can be physically or logically separated from each other. The separate KBS are integrated in and managed by only one DKBMS. This is illustrated by Figure 1. Each KBS is made up of devices for implementing an inference mechanism, DB, KB and a user interface. The knowledge base management system (KBMS) is a mechanism, and a language that provides access, as well, to the knowledge base(s) and the database(s). There can also be direct access to the DB, and it is provided by either the database management system (DBMS) or the distributed database management system. Every KBS can be implemented by using a different KBMS.

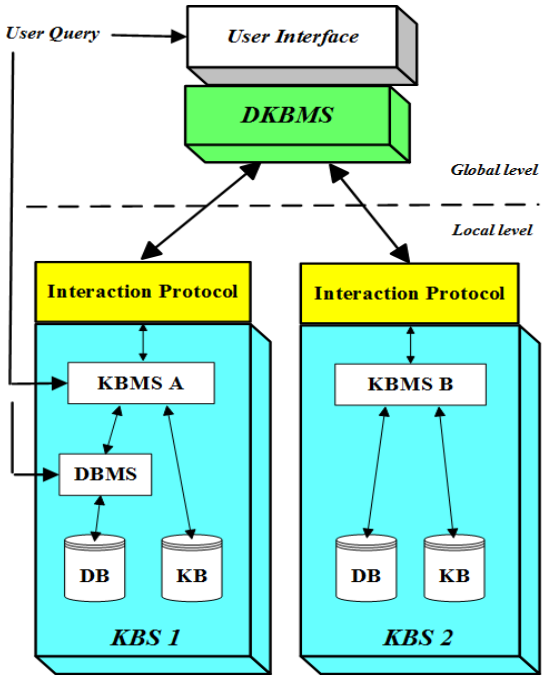

Figure 1. General structure of Distributed Knowledge-Based System

A DKBMS has three components, namely a global KB, a knowledge cluster dictionary and an interaction protocol (see Figure 2). The global KB incorporates knowledge describing the overall system organization and meta-knowledge, determining the scope of knowledge that is managed by each individual KBS.

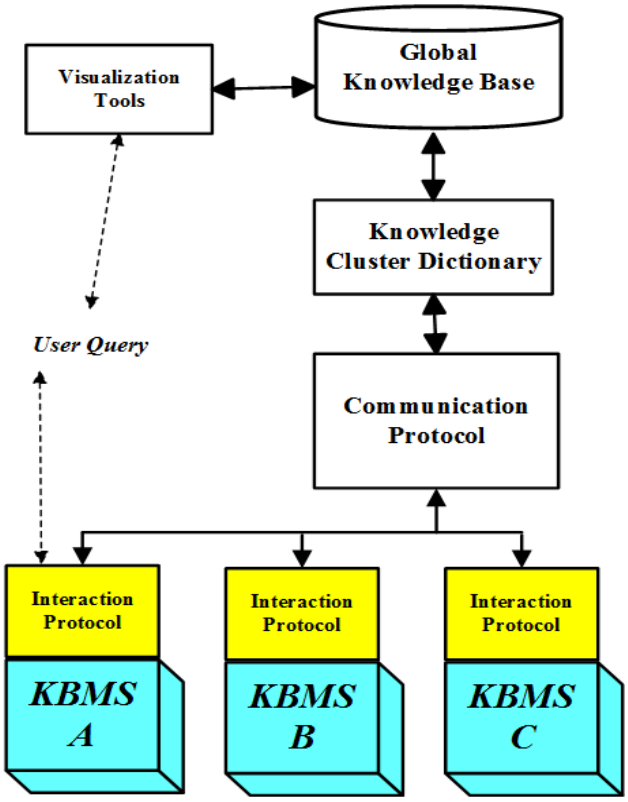

Figure 2. Interaction among Distributed KBMS components

User queries can be directed either to the DKBMS or to a particular, specific KBS. When the user puts a query to DKBMS, the global KB uses the knowledge cluster dictionary to determine which local DBs, if there are any, it holds the knowledge needed for performing the task defined through the query. Several local KBSs can cooperate with the global KB to complete the query. The global KB has the necessary meta-knowledge for selecting the local KBS which are most likely to achieve the goal set by the user.

The second component of the architecture is the knowledge cluster dictionary. It categorizes the sub-goals of the global KB in logically connected groups. These sub-goals are derived from the initial user goal and the global KB meta-knowledge. The knowledge groups are determined as a part of the project in conjunction with particular DKBMS application. The dictionary consists of two elements: assigning the global KB sub-goals to a specific knowledge group and assigning each knowledge group to one or more local KBSs. When a knowledge group is assigned to a local KBS, the sub-goals are also assigned to this KBS. It is assumed that the particular KBS contains knowledge needed for achieving these sub-goals. The architecture chooses the unnecessary knowledge among the individual locations by distributing the multiple locations.

The third component of the DKBMS involves the communication relationship between the DKBMS and the local systems. The local KBSs interact with the DKBMS through a standard message protocol and the local locations use the DKBMS as a link, which either gives instructions or makes queries to other local KBSs. Heterogeneous knowledge representations are supported in the local KBS by a common interaction protocol, and a correspondence functions between the global KB representation and each local KB representation.

The main difference between interaction protocols and communication protocols lies in [1], [10]: (1) the knowledge exchanged between the agents is of a higher level of abstraction, not just separate pieces of information, and (2) the interaction protocols may describe different strategies of problem solving such as cooperation, negotiation, etc. The interaction protocols are frameworks which restrict the possible interactions between the agents with the aim of achieving a certain goal. The interaction protocols can be regarded as transition networks representing the states acquired by the agents in the course of the interaction. An agent takes one state or another depending on the type of message it gets and also on the condition met by the agent in other to traverse a link. Such types of graphs are used for formalizing the known protocols.

A great advantage of this three-layer architecture is the fact that it provides logical separation of the DKBMS and the local KBSs. The conceptual model of this architecture can be implemented in any paradigm used for knowledge representation. The outlined architecture is suitable for domains in which:

- The problem is externally distributed, i.e. the individual components are logically and geographically dispersed;
- The agents, responsible for making decisions, are semi-autonomous;
- The atomic data is not an adequately representation for the purposes of modeling the agents. Therefore, it is necessary to apply the paradigm using a KB;
- The requirements for representing the knowledge of individual agents are not generally valid. The size, complexity, determining characteristics and stability of agents vary. Thus, heterogeneity is a preferable condition at the model location level;
- It is preferable to achieve coordination and compatibility of the global system.

Therefore, the distributed, semiautonomous, intelligent entities should cooperate with each other to solve their problems and conflicts.

3. Generalized implementation of the system

An object-oriented approach is used for developing the DKBMS. This method is chosen because it represents a natural way of representing information through frames. It also supports the heterogeneous knowledge presentations in local KB, and facilitates the communication protocol which links the components of the descried architecture.

The primary classes of the suggested prototype are shown on Figure 3. The distributed system is managed by the GKB and DKBMS classes which inherit the *Object* class properties from the standard container class library. The *ClusterDict* and *LanguageDict* classes manage the knowledge cluster dictionary and the dictionary of language matches. The *Frame* class determines the requirements to the meta-knowledge within the framework of the global KB, while the *Slot* class determines the slots within the framework of each object that belongs to the *Frame* class. The *Cluster* class describes the content of each knowledge group which is used for organizing the sub goals within the knowledge group dictionary. The suggested inheritance class hierarchy uses the *Dictionary* class from the standard container library of classes. The *Inference*, *Assertion* and *Rule* classes describe the meta-knowledge which is stored in the frame slots of the GKB class.

Figure 4 provides an example illustrating the object interaction. The frames of the GKB class are used for organizing and planning meta-knowledge utilized by

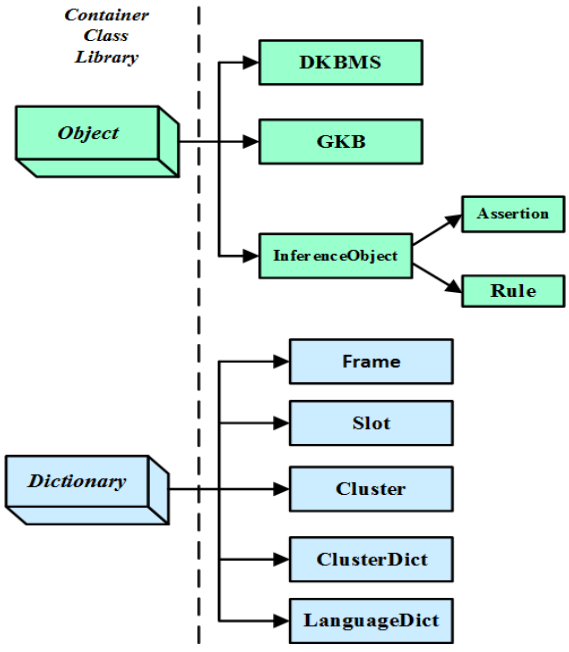

Figure 3. Distributed KBMS class structure

the *inferSubGoal* method which performs locking based on sub-goals. One or more sub goals will be derived from the process for storing meta-knowledge in the frame slots while the inferences are being processed. These sub-goals are gathered in knowledge groups as a result of the knowledge design process. The relations between the groups and sub goals are supported by the *Cluster* class objects which are assigned to the particular KBS through the *ClusterDict* class objects. Regardless of the ways in

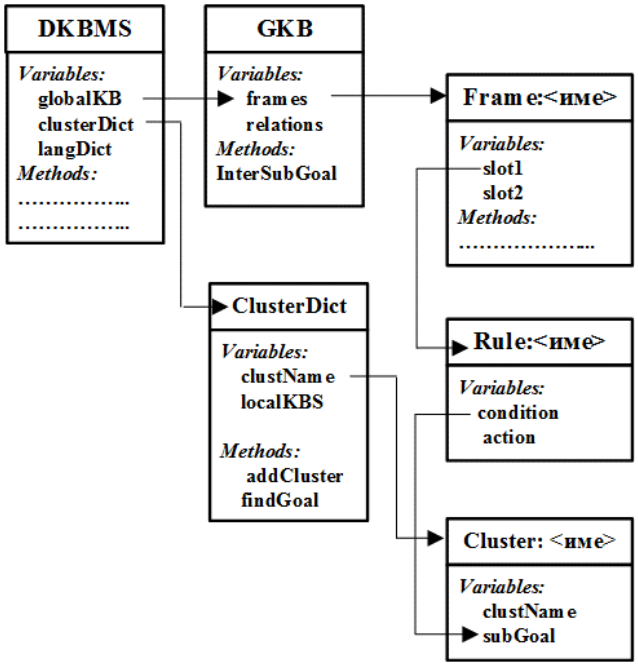

Figure 4. Interaction among DKBMS objects

which knowledge is represented, the particular knowledge management systems should provide a mechanism for sending and receiving messages that are consistent with a standard protocol.

Object-oriented systems are suitable for the implementation of distributed processing because the data flow and management are carried out through exchanging messages between the objects. An appropriate mechanism for addressing these messages is provided, so the operations on the objects can be activated without considering their locations. This allows moving the objects during the performance and, therefore, creating opportunities for dynamic reconfiguration. The distributed object-oriented paradigm provides a complete syntactic and semantic equivalence of local and remote initiation in which the object calls are transferred as arguments.

4. Implementation of the interaction protocols

The aim of this section is to propose certain architecture of a community regarding interacting intelligent agents which collaborate in solving a posed problem. As presented above, the basic architecture of a sole intelligent system comprises a knowledge base, an inference engine and a user

interface module. The process of problem solving in a rule-based system starts with setting a global goal which is to be achieved in the course of work, and may give rise to a number of sub-goals. This kind of inference mechanism is applied in both data-driven and goal-driven strategies and implies the implementation of the system with distributed artificial intelligence. The architecture of the multi-agent distributed system is proposed by the authors, and it is presented in Figure 5.

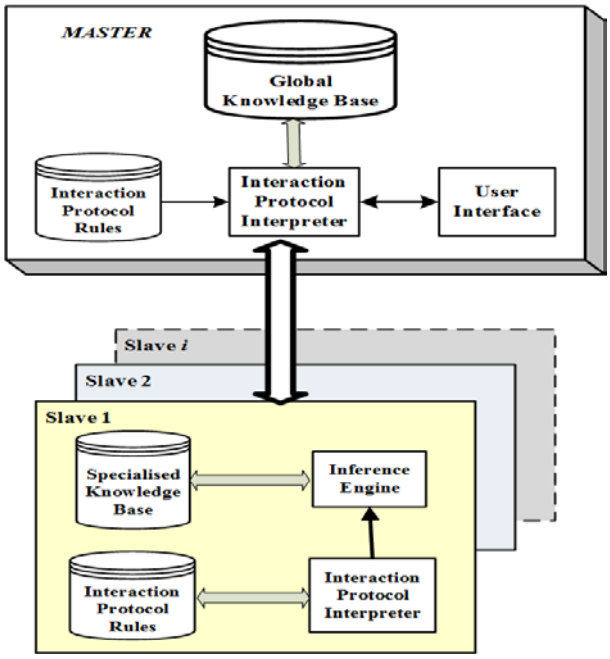

Figure 5. Interaction between master and slaves

This includes a master process and multiple autonomous processes – agents, each of them specialized in a subdomain of the problem being solved. This specialization implies distribution of the knowledge base. The agents may as well collect on-line information related to the current problem. The architecture of each agent (slave) is expanded with a set of production rules, determining the interaction protocol and an interpreter of those production rules.

The role of the master agent is to start the protocol in order to present the global goal, and to perform the user interaction. The structure and the type of protocol message are: REQUEST, INFORM, REJECT, ACCEPT, NOOPINION. The general protocol of interaction between the autonomous agents is presented in Figure 6. *S1, S2, S3, S4, S5, S6* are the states through which the agent are passing in the course of the dialog. *S4, S5, S6* are end states, terminating the interaction. The protocol is presented by production rules, as it is described above. For instance, the following rule describes the transition from S1 to S2:

```
If state S1 & message is "propose"
    then state S2
```

The protocol is started by the master agent being in state S1, with a message REQUEST which is normally broadcast to all the remaining slave agents. Through this message the master proposes a solution hypothesis (goal) to be checked by the specialized agents. In response, the receiving agents start the inference engine over their local knowledge base. In the course of the work, the agents evaluate a confidence factor for the hypothesis being proposed. Based on this value they may confirm the hypothesis with the estimated confidence factor upon which INFORM the master about the results obtained or to

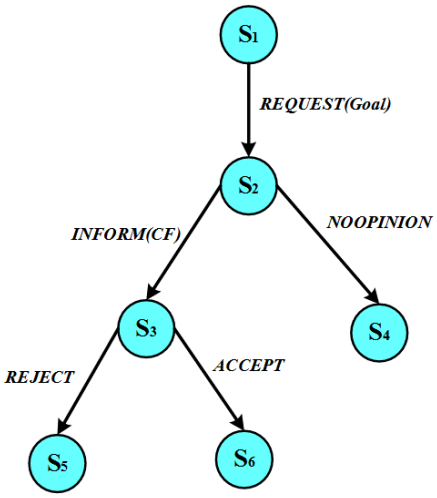

Figure 6. Graph representation of the interaction protocol

announce that the information available to them is insufficient for a decision to be taken (NOOPINION).

The master agent based on the information gathered from the slaves may decide that the hypothesis is to be accepted, upon which an ACCEPT message is to be broadcast, or to be rejected and the proper response will be sent. In fact, the conditions of the state transitions constitute the message type obtained.

A diagnostics expert system was developed. Its knowledge base gives the relations between symptoms and defects for a number of tractor engine models. The knowledge base is represented by rules, the preconditions stating the presence or absence of a given symptom, while the conclusions define the possible defects which have caused the symptoms. Thus, each symptom is related to a number of different defects and the strength of the relation is evaluated through confidence factors which are the product of the expert's long accumulated experience in the field. An example rule:

```
IF formation of foam is observed
   in the reservoir and lubricant
   leakage is detected
THEN it may be asserted that with
   a confidence of 40%,
   air is sucked in the system,
OR with 54%, there isn't enough
   lubricant in the reservoir,
OR with 37%, the hydraulic
   pump has defected,
OR etc.
```

The system has a built-in plausible inference engine which provides incremental evaluation of the confidence factor. In this way the engine accumulates and reinforces the confidence in a given solution. The knowledge base is distributed among the autonomous agents, so that each of them is specialized in the area of certain symptoms. The solution of the problem regarding determination the possible cause of the engine malfunctioning is achieved through the following stages, as specified by the interaction protocol shown in Figure 6:

1) Each agent retrieves a parallel solution in its local knowledge base, independently of the other agents. A confidence factor (CF) is estimated for the solution achieved. A predefined threshold value for the CF determines the assessment condition for the protocol – whether to INFORM the master on the resultant CF obtained or to send NOOPINION message in case of CF less than the threshold value.

2) Based on the information in its knowledgebase the master agent combines and merges the partial solutions to produce a global one. The master agent then informs the agent on the final solution.

The interaction protocols are a means of:

(1) Distributing the knowledge base and the inference procedures in a knowledge-based system;

(2) Flexibly specifying the communications between the agents through the protocol definition rules and the interpreter of those rules.

5. Conclusion

The distributed knowledge-based system architecture requires integration of knowledge and data. The authors suggest the use of KBMS that manages the integration of knowledge and data in each location. This architecture provides direct access to the DB of the local KBS through the existing DBMS. If the DKBMS requires information from the DB, while the global aim is being achieved, it uses the interaction protocol to send a message to the local KBMS responsible for the required data. If the data is also distributed, the KBMS communicates with the DDBMS in order to retrieve the data. The appropriate definition of the knowledge clusters and the distribution of the local KBS to these groups affect significantly the execution and flexibility of the particular applications that utilize this architecture.

The offered architecture of the distributed knowledge-based system guarantees the integration of knowledge and data within the framework of the global system, and also allows the heterogeneous knowledge-based systems to communicate through the management system of the distributed knowledge base. This architecture can be used for developing information systems which support a distributed group of semi-autonomous units that make decisions. This group decision making process requires interaction between the DB and KB by coordinating the communication between them.

The suggested approach may be applied to co-operative distributed knowledge-based systems intended to be assistants rather than autonomous problem-solvers. The major concern with those is to determine how two agents which have incomplete and overlapping knowledge can co-operate on a task.

References

[1]. Calvaresi, D., Appoggetti, K., Lustrissimini, L., Marinoni, M., Sernani, P., Dragoni, A. F., & Schumacher, M. (2018). Multi-Agent Systems' Negotiation Protocols for Cyber-Physical Systems: Results from a Systematic Literature Review. In *ICAART (1)* (pp. 224-235).

[2]. Juneja, D., Dhiman, C., Monga, S., Singh, A. (2018). Compendious study of interaction protocols in multi-agent systems. *International Journal of Engineering & Technology*, *7*(3.8), 1-6.

[3]. Kirkman, D. (2016). A distributed knowledge approach to managing innovation. *Journal of Strategic Innovation and Sustainability*, *11*(1), 9-15.

[4]. McCreary, D. & Kelly, A. (2014). *Making sense of NoSQL*. Manning Publications.

[5]. Coulouris, G., Dollimore, J., Kindberg, T. (2011). *Distributed systems: concepts and design*. Pearson.

[6]. Fotache, G. (2013). Comparative Study regarding information management and knowledge management. *Economy Transdisciplinarity Cognition*, *16*(2), 63-70.

[7]. Mohajan, H. (2017). The roles of knowledge management for the development of organizations. *Journal of Scientific Achievements*, *2*(2), 1-27.

[8]. Halawi, L., McCarthy, R., Aronson, J. (2017). Success stories in knowledge management systems. *Issues in Information Systems*, *18*(1), 64-77.

[9]. Özsu, M. T., & Valduriez, P. (2011). *Principles of distributed database systems*. Springer Science & Business Media.

[10]. Flor, M., Yoon, S. Y., Hao, J., Liu, L., & von Davier, A. (2016, June). Automated classification of collaborative problem solving interactions in simulated science tasks. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 31-41).

[11]. Marz, N., & Warren, J. (2015). *Big Data: Principles and best practices of scalable real-time data systems*. New York; Manning Publications Co..

[12]. Weinreich, R., & Groher, I. (2016). Software architecture knowledge management approaches and their support for knowledge management activities: A systematic literature review. *Information and Software Technology*, *80*, 265-286.

[13]. Karnavas, Y. L., & Chasiotis, I. D. (2016). A simple knowledge base software architecture for industrial electrical machine design: application to electric vehicle's in-wheel motor. In *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology–ISAT 2015–Part IV* (pp. 111-122). Springer, Cham.

[14]. Li, Z., Liang, P., & Avgeriou, P. (2013). Application of knowledge-based approaches in software architecture: A systematic mapping study. *Information and Software technology*, *55*(5), 777-794.