Application of an integration platform for ontological model-based problem solving using an unified semantic knowledge representation

Valerian Ivashenko
Department of Intellectual Information Technologies
Belarusion State University of Informatics and Radioelectronics
Minsk, Republic of Belarus
ivashenko@bsuir.by

***Abstract*—This article describes a solution in the form of an intelligent integration platform based on the model of an unified semantic knowledge representation for the development of applied knowledge-driven systems. The model of an unified semantic knowledge representation using semantic networks, models and methods of measure and probability theories, methods of descrete optimization and applied mathematics, computer simulation and multi-agent approaches were used. The purpose is to develop computer tools with cognitive architecture relying on elements of artificial consciousness and being able to communicate and to be flexible and adaptive in complex educational applications. Virtual machines, subsystems of integration platform and tutoring applied multi-agent software system were developed and implemented as part of human-machine interaction system.**

***Keywords*—artificial intelligence systems integration, integration platform, multi-agent system, knowledge-driven system, knowledge processing model, unified semantic knowledge processing model**

I. Introduction

The basis for the success of a learning intellectual system is integration openness [42], [45]. When it is possible to integrate not only new knowledge related to different types but also various mechanisms for solving problems. Integration is one of the important understanding mechanisms for knowledge systems allowing the acquisition and improvement of problem-solving skills which is important for knowledge-driven systems [42], [45]. This article presents a solution in the form of an integration platform based on the unified semantic knowledge representation model. This presentation attempts to answer the following questions.

1 What goals can be achieved using such platforms?
2 What developments are already in this direction?
3 What are the architecture, mechanisms and rules for using the platform?
4 What are the positive and negative peculiarities of the platform?
5 What results have been achieved in the process of its application?
6 What are the perspectives for the development of the platform?

The general goals planned to be achieved are:

- creation of dynamically updating knowledge-based system able to accept new knowledge via machine learning and high-level or natural language communication;
- creation of scalable knowledge-driven systems maintaining big knowledge and large scale integrated ontology;
- creation of artificial consciousness systems which are self-descriptive and introspective;
- creation of multi-agent distributed applied intellectual systems.

II. Overview of models and approaches

The necessity of artificial intelligence integration historically caused by the existence of separate artificial intelligence solutions for specific problems such as reasoning knowledge, speech synthesis and recognition, computer vision problems. General approaches to the integration of information systems: integration through translation with control passing, integration through interpretation or through communication without control passing. In the case of control passing each process or agent of system should store own state. Therefore, the corresponding storage is available for one control flow and both are shared by all processes or agents on the system.

From the other side, consciousness as a more advanced kind of intelligence is determined by social experience of natural language communication. Thus, by natural way, such communicative models as actor models [1], [5], [13] or multi-agent systems are the basis [6], [45] not only for concurrent computer system but artificial intelligence systems. Therefore, concurrent models are integration models. These models are divided into two classes. The first class is models without shared common memory or storage. The second class is models with shared common

memory or storage. Such models and means as actor models [5], Communicating Sequential Processes [10], Calculus of Communicating Systems [12], pi-calculus [15], Algebra of Communicating Processes [14], Language of Temporal Ordering Specifications [16]–[18], reactive multi-agent systems [13] and so on [19]–[23] can be referenced with the first class of models. The models of the first class using message passing communication provide implementation basis of the collective or swarm intelligence.

Models of the second class is oriented to maintain shared storage. Shared storage is seemed to be capable to maintain large knowledge bases and ontologies to solve complex problems and to provide sophisticated processes control. The bright examples of such approach are blackboard architecture models and blackboard systems. Blackboard system is shared information resource with the own communication protocols and
or consistency model. Means oriented to use with blackboard system are: CORBA, MOSID, OpenAir, OAA and others [32], [34]–[36].

MOSID (Messaging Open Service Interface Definition (OSID)) is an Open Knowledge Initiative specification which provides a means of sending, subscribing and receiving messages. OSIDs are programmatic interfaces which comprise a Service Oriented Architecture for designing and building reusable and interoperable software [35].

CORBA is a standard can be used for AI systems integration [32]. CORBA enables software components written in multiple computer languages and running on multiple computers to interoperate. CORBA is developed by the Object Management Group. Similar standard developed by Microsoft was DCOM (Dynamic Common Object Model).

OpenAIR Protocol is a routing and communication protocol based on a publish-subscribe architecture. It is mean and environment («AIR») that allows numerous A.I. researchers to share code more effectively. This is mean for distributed multi-module systems. OpenAIR is oriented to be foundation for markup languages and its semantics of hardware-software interfacing including computing vision (as at CVML (Computer Vision Markup Language)), gesture recognition and generation and so on. OpenAIR Protocol follows similar principles and architecture as the CORBA.

OAA (Open Agent Architecture) is a hybrid architecture that relies on a special inter-agent communication language (ICL) [36]. ICL is a logic based declarative language adopted to express high-level, complex tasks being close to natural language expressions.

There are implemented systems which use models of the second class. Psyclone AIOS is an implementation of a blackboard system that supports the OpenAIR message protocol [34]. It is a software platform, or an AI operating system (AIOS), developed by Communicative Machines Laboratories for use in creating large, multimodal artificial intelligence systems. Elvin is a content-based router with a central routing station, similar to the Psyclone AIOS [33].

Examples of applied integrated systems based on this approach are such robots and humanoids as MIRAGE, ASIMO, QRIO, Cog, AIBO and etc.

The important aspect of models is possibility of process introspection including methods of process mining for the purpose of inductive programming using action languages. Such languages as LTML (Learnable Task Modeling Language), PDDL (Planning Domain Definition Language) [7]–[9] and MAPL (Multi-Agent Planning Language) [6] can be considered as means of plan specification of artificial intelligence systems including concurrent and multi-agent systems.

The next approach can be viewed as pragmatic or problem specific approach. This approach concentrates on developing cognitive architectures. Examples of cognitive architectures are: 4D-RCS (Real-time control system) Reference Model Architecture [29], SOAR (State Operator And Result) cognitive architecture [30], architecture of Hierarchical temporal memory [31] and many others. The models investigated in the range of OSTIS projects can be also referred to this last approach [41].

Another approaches and models are determined by history of development of computing systems for artificial intelligence. These include architectures of LISP-machines (Connection machines), PROLOG-machines and machine learning processors and accelerators.

However, these architectures and models have restricted capabilities of knowledge integration via unified representation including limitations to deal with NON-factors of knowledge [42], [48], to introspect processes using semantic logging and to combine knowledge declarative semantics with operational semantics of synchronous and asynchronous knowledge processing using various knowledge representation languages.

III. Integration platform description

Proposed integration platform mainly concentrated on integration via translation and via communication. While integration via interpretation limited by languages of unified semantic knowledge representation model and languages supported by model used to implement this platform.

Specification is main part of self-descriptive and introspective systems such as knowledge-driving systems. There are several models and means to specify and implement such discrete systems as abstract machines or information processing models including concurrent systems. These are transition systems, actor models, Communicating Sequential Processes, Calculus of Communicating Systems, pi-calculus, Algebra of Communi-

cating Processes, Language of Temporal Ordering Specifications, temporal logics and others [1]–[4], [11]–[15], [17], [18], [28], [42]. Proposed knowledge specification model [42] is the key mean to specify knowledge represented with the unified semantic knowledge representation model [42], [45].

Knowledge processing model [42] describes dynamic of knowledge accumulation and optimization processes. To define knowledge processing model seven components need to be specified: alphabet, language, syntactic relations, initial states, interpretations, operations and semantic metric [42]. Every finite structure can be specified with its formal ontology model. Languages are specified by formal ontology model relations mapping its texts into their representations in its other texts and by the relation between formal ontology models of allowed and forbidden syntactic structure and formal ontology models of representations of language texts in its other texts. Initial states are specified as sublanguages. Infinite number of initial states can also be specified by the knowledge specification models relation which specify (generative) initial states order with a finite number of primary initial states (information constructions or language texts). Language syntactic (incidence) relation is specified as a language via its texts representations. Language semantics specification consists of denotational semantics and operational semantics specifications [1]–[4], [13]. Denotational semantics is specified by the knowledge specification model relation between formal ontology models of language text fragments and formal ontology model of finite subsets of its denotations. Such relation specify whatever mapping which is projection of interpretation of language text fragments on finite sets of denotations. Operational semantics is specified by the knowledge specification model relation between formal ontology models of reifications of projections of language text fragments interpretations on finite sets of denotations in two situations. In case of reflexive semantics, denotation semantics specification can express syntactic restrictions for forbidden syntactic structures in the vicinity of language key elements. The allowed syntactic structures are all which are described in denotational semantic specifications, any other structures are forbidden. Thus, we get specifications of language key elements. Semantic metric can be specified by mapping language texts to metric space. Simple semantic metric is specified by mapping language texts to binary logic scale σ with exclusive disjunction as a metric operation.

$$\sigma \in \{\Lambda\} \times \{\bot, \top\}^{\Lambda} \times \{\{\bot, \top\}\} \tag{1}$$

$$\psi\left(\sigma_2(\alpha) \veebar \sigma_2(\beta)\right) \tag{2}$$

$$\psi(\chi) \underset{def}{=} \begin{cases} 0 \,|(\neg\chi) \\ 1 \,|\chi \end{cases} \tag{3}$$

The equivalence can be also considered as a semantic metric operation due the isomorphism existence between equivalence and exclusive disjunction.

$$\psi\left(\sigma_2(\alpha) \veebar \sigma_2(\beta)\right) = 1 - \psi(\sigma_2(\alpha) \sim \sigma_2(\beta)) \tag{4}$$

More sophisticated metrics may take into account syntactical and spatial-temporal structure of knowledge.

One knowledge processing system integrated by the another knowledge processing systems if some conditions fulfilled [27], [42], [46], [47]. The necessary conditions to integrate one knowledge processing system in the second is the existence of such text inclusion mapping π and bijection [27] i as

$$\begin{array}{l} \forall\rho\exists i(\rho)\left(\rho = i \circ \pi \circ i(\rho) \circ \pi^{-1} \circ i^{-1}\right) \\ \forall\rho\exists i(\rho)\left(\pi^{-1} \circ i^{-1} \circ \rho \subseteq i(\rho) \circ \pi^{-1} \circ i^{-1}\right) \\ \forall\rho\exists i(\rho)\left(i(\rho) = \pi^{-1} \circ i^{-1} \circ \rho \circ i \circ \pi\right) \\ \forall\rho\exists i(\rho)\left(i \circ \pi \circ i(\rho) \subseteq \rho \circ i \circ \pi\right) \end{array} \tag{5}$$

where π is a text mapping relation between text fragments and texts of the second knowledge processing model containing its; i is bijective integration mapping; ρ is an operation.

These can be shown with the next diagram.

$$\begin{array}{ccccc} \sigma & \overset{i}{\longleftrightarrow} & i(\sigma) & \underset{\leftarrow}{\overset{\pi}{\rightarrow}} & \tau \\ \rho\downarrow & & \overset{i}{\longleftrightarrow} & & i(\rho)\downarrow \\ \omega & \overset{i}{\longleftrightarrow} & i(\omega) & \underset{\leftarrow}{\overset{\pi}{\rightarrow}} & \gamma \end{array} \tag{6}$$

Architecture of implemented system is based on architecture of control levels for knowledge-driven systems [42]. These levels are: device control level, data control level, knowledge control level. The levels of control are disposed along the implementation hierarchy direction (vertical). While there are several sub-architectures which are placed along the communication direction (horizontal). These sub-architectures relate to abstract machines or information processing models which correspond to different variants of implementation subplatform. There are two virtual machines which implements core of the proposed integration platform: variety virtual machine and ontology virtual machine [37], [42], [45].

If an implementation is considered as a problem domain then the objects of the implementation are parts of the problem domain of this implementation. These objects, its kinds with the relations between them form a subject domain of the implementation. Any finite part of the subject domain of the implementation can be specified accordingly with the knowledge specification model. Various models are being specified depending on which objects are included in a corresponded part. Kinds of such models are situation structure model, system structure model, motion model, process model, device structure model, device process model, instruction

set syntax model, instruction set semantic model, typed data structure representation model, typed data structure process model, typed knowledge structure syntax model, typed knowledge structure process model. From the point of view of architecture of a chosen implementation these models can be grouped in more complicated models. A distributed system can be represented with a set of communicating nodes. Each node can be realized as a computing machine. A machine realizes two or three levels of control. Machines encapsulate knowledge, data, and such devices as processors, memory and controllers. Each subsystem of a machine is specified with a model. Models are specifications of subsystems of implementation. Memory models, operation models, allocator memory models, synchronization (activation
deactivation) models, access management models, reallocation memory models, number and strings processing models are more specialized models. Models of machines can share or include each of them [42], [45].

The platform was implemented with the virtual machines of two types: ontology virtual machine (knowledge processing machine) and variety virtual machine (user interface data processing machine) [37], [45]. The ontology virtual machine shares all three levels of control while variety virtual machine implements first two of them.

Subsystems of the ontology virtual machine specified with memory, allocator, allocation and reallocation, machine operation instruction subset, synchronization (including excitation and inhibition), access management, strings processing, unified semantic processing, communication and IO controller models [42], [45].

Subsystems of the variety virtual machine specified with memory, allocator, allocation and reallocation, machine operation instruction subset, strings processing, communication and IO controller models [37], [45].

All knowledge structure models match the unified semantic representation model [42].

All knowledge programming models (unified semantic processing) match knowledge processing model based on the unified semantic representation model. Operations of ontology virtual machine instructions are based on string processing model for knowledge-driven systems. They are operation both with such simple data types as numbers and also with strings. Strings are used as multiple nested stacks or meta-stacks for ontology virtual machine [42], [45].

Programs can be structured using sc -chains or sc -sequences. The lasts differ from the first by link membership connectives which not include a membership of contained element of the next link but include a membership of the next link to sequence set which contains all links and the membership of the first link. The structure of each instruction (command) or operator shares the METAPHORM [44] principles including the ordered pairing of input and output parameters and the encoding of operator types using either associative positioning (using key elements) or a structural morphological approach. The operator semantics of basic task types can be defined with systems of patterns (similar to semantic query language constructions [43]) for search and construction operators and sets of deleting elements for delete operators. Commands can be constructed during the processing of SEC (Semantic Execution Code) texts or their representations by sc -chains [42], [45].

Ontology virtual machines integrates operations with not only asynchronous but synchronous semantics. Synchronization is implemented by mechanism of inhibition and excitation phases. Thus, processes of the next excitation phase can not be started before finalizing all synchronous processes of previous excitation phase. Some processes can be triggered and suspended during inhibition phase. All ready results of each operation can be represented with an actual membership to a sc-set of ready results [42]. Thus, all events can be reduced to membership events. Processes generating these events are triggered by mechanism of inhibition and excitation phases.

Synchronous and asynchronous semantic specification is constructed using phenomenons description ontology for key elements with operation semantics and their semantic vicinities.

Knowledge access and control model also can be considered part of inhibition phase mechanism. Knowledge access and control model deals with agents, areas, modes and its types of access.

Multi-agent interaction model includes active structure reconfigurable (Semantic Code) memory maintaining synchronization mechanisms and models for semantic logging and multi-agent plan specifications languages [42]. The common interaction scheme for active structure reconfigurable memory is show on the figure below.

Multi-agent interaction is represented by events and relations between them in forms of phenomena, protocols and plans [42]. The spatial and temporal relations should be used to define spatial and temporal constraints for the plans. Unlike to MAPL constructions [6], these constrains do not use topology of the line of real numbers and the corresponding scale but the topology of events with the corresponding scale of rough sets over the lattice of their sets. Consistency of a multi-agent plan is determined by the several requirements. These are:

$$(N(m) \cap P(n)) \cup (P(m) \cap N(n)) = \emptyset \qquad (7)$$

where m is a membership event, while n is a non-membership event between same elements, N and P are lower and upper bounds membership functions of L-fuzzy rough sets (sc -set) [42]. In the case of defined clock measure μ and time duration δ consistency is

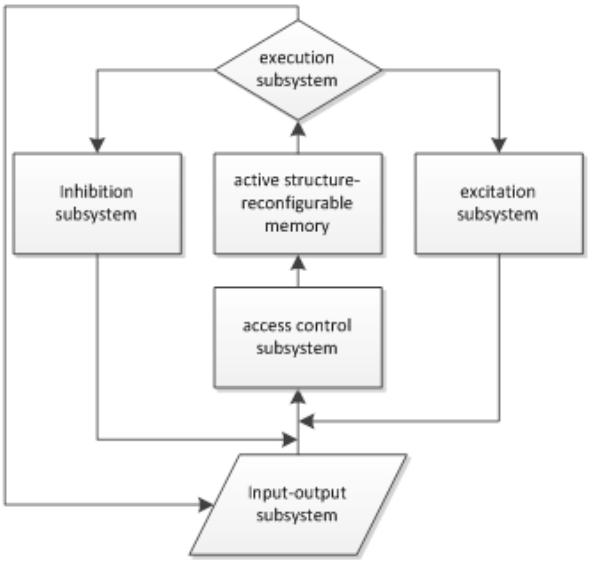

Figure 1. Scheme of subsystem interaction.

determined by the following requirements:

$$\begin{gathered}\delta(\emptyset)=0\\ \delta(S\cap T)\leqslant\delta(S)\\ \delta(S)\leqslant\delta(S\cup T)\\ \delta(S\cup T)\leqslant\delta(S)+\delta(T)\\ ((S\prec\!:T)\wedge(S\triangleright\triangleleft T))\rightarrow(\delta(S\cup T)=\delta(S)+\delta(T))\end{gathered}\tag{8}$$

where $\prec\!:$ is temporal precedence relation, $\triangleright\triangleleft$ is disjoint relation and S and T are phenomenons.

There are two general class of agents: internal agents operating only with active structure reconfigurable memory and external agents communicating via IO controllers or operating with variety virtual machine. More detailed specification of variety virtual machine is available at [37].

The general principles of knowledge processing [42] for the integration platform are:

- accounting of NON-factor;
- semantic logging;
- knowledge stream processing.

IV. Other approaches: competition and cooperation

Despite the universality of proposed integration platform and its knowledge representation capabilities, there are performance limitations as for its implementation as for a implementation of any other computer architecture, abstract machine or information processing model [42]. That is why there is a number of systems which can not be effectively implemented on the current implementation of the integration platform. These systems can be implemented separately and be able to concurrent with the platform in cooperative or competitive mode. The communication between integration platform and such external separative system is organized by the programming interfaces provided by the input-output subsystems (IO controllers) of the integration platform.

V. Application of integration platform

Ones of the most common classes of generalized problems are the searching, the choosing, the verifying, the constructing, the reconstructing and the destructing, while the searching, the constructing and the reconstructing are the most common classes of individual problems [42]. Problems of all classes can be composed using problems from the searching, the reconstruction and the destruction classes. There are agents for problems of each class [45]. Agents solving any cognition problem, i.e. search, choice or verification, are cognitive agents. Call the other agents performative agents. Depending on correspondence type between agent states and states of whole knowledge base these agents can solve external or internal problems. Therefore, there can be external and internal agents for searching, choosing and verifying. It is important to admit that an agent who solves the internal problem of search or another cognition problem may not be able to solve any cognition problem as an external problem. In the process of human-machine interaction tasks of different kinds arise. Consider the tasks arising in intelligent tutoring systems. Such systems are able to provide students with educational material in the reference system mode, from the other side there is a task to examine student knowledge. In the reference system mode there are three types of problems: wait user question (waiting is kind of the searching), retrieve information and output the result to the user. These problems can be solved correspondingly by agents of three types: external and internal search agents and external performative agent. All these agents solve task of navigation on tutorial materials. During navigation human-computer dialog contains interfaces with suggestions to user which can be interpreted as alternative questions. Each user reaction can be interpreted as answer on such questions. During a series user reactions on independent suggestions quantity of received information can be calculated as:

$$\left\lceil\left(\sum\nolimits_{i=1}^{q}\log_2\left(v\left(i\right)\right)\right)-\sum\nolimits_{i=1}^{q}\log_2\left(t\left(i\right)\right)\right\rceil\tag{9}$$

where $t\left(i\right)$ is number of confirmed variants among $v\left(i\right)$ different choice variants on suggestion i. When the suggestions are causally dependent, received information volume can be calculated as:

$$\left\lceil\log_2\left(\sum\nolimits_{i=1}^{q}\left(\frac{v\left(i\right)}{t\left(i\right)}-t\left(i\right)\right)\right)\right\rceil\tag{10}$$

When the suggestions are dependent and alternative, received information volume can be calculated as:

$$\left\lceil\log_2\left(\sum\nolimits_{i=1}^{q}\left(\frac{v\left(i\right)}{v\left(i\right)-t\left(i\right)}+t\left(i\right)-v\left(i\right)\right)\right)\right\rceil\tag{11}$$

Accordingly the unified semantic knowledge representation model all knowledge at the knowledge control level are represented as homogeneous semantic networks [42]. Therefore, tutorial material accessed by intelligent agents is represented by semantic network too. This material can contain others types of data such as natural texts, graphic images, audio files. Thus, used semantic network can be classified as a hypermedia semantic network [40].

The other problem solving by intelligent tutoring system is to examine student knowledge and compute its measures (scores) [45]. There are also three types of agents: external search and performative agents and internal knowledge measurement performative agent. The measure function can be expressed by following formulas:

$$m\left(\frac{\max\left(\{0\}\cup\{\rho\left(q\right)*\pi\left(q\right)-q*\sigma\left(q\right)\}\right)}{q*\left(\pi\left(q\right)-\sigma\left(q\right)\right)}\right) \quad (12)$$

where q is number of questions. Each of them has $v\left(i\right)$ risk chances to answer. Also,

$$\begin{aligned}\pi\left(q\right)&=\prod_{i=1}^{q}v\left(i\right)\\ \sigma\left(q\right)&=\sum_{j=1}^{q}\frac{\pi(q)}{v(j)}\\ \rho\left(q\right)&=\sum_{j=1}^{q}r\left(j\right)\end{aligned} \quad (13)$$

$$\kappa\left(q\right)=\frac{\sigma\left(q\right)}{\pi\left(q\right)}, \quad (14)$$

$$m\left(x\right)=10*x. \quad (15)$$

To reduce the range of computed values $\pi\left(q\right)$ can be computed as least common multiple.

$$\pi\left(q\right)=\begin{cases}LCM\left(\{v\left(q\right)\}\cup\{\pi\left(q-1\right)\}\right)|q>1\\ v\left(1\right)|q=1\end{cases} \quad (16)$$

If a question i has $t\left(i\right)$ right and $f\left(i\right)$ wrong alternative homogeneous answers then

$$v\left(i\right)=\frac{t(i)+f(i)}{t(i)} \quad (17)$$

If right answer is one of the question answers which are short strings in alphabet having a symbols with the length restricted from s to h then

$$v\left(i\right)=\sum_{k=s}^{h}a^{k} \quad (18)$$

If unique right answer has length $l(i)$ while question answers are medium strings in alphabet having a symbols with the length restricted from s to h then

$$v\left(i\right)=a^{l(i)}*\left(h-s+1\right) \quad (19)$$

If unique right answer has length $l(i)$ while question answers are strings in alphabet having a symbols [39] with the length restricted from s to h then

$$v\left(i\right)=\left(2*a\right)^{l(i)}*\sum_{k=s}^{h}2^{-k}=\left(2*a\right)^{l(i)}*\left(2^{1-s}-2^{-h}\right) \quad (20)$$

If two series of questions (with q_i and q_k questions) have no dependent questions then joint series has q_j questions which satisfy following expressions

$$\begin{aligned}q_j&=q_i+q_k\\ \pi\left(q_j\right)&=\pi\left(q_i\right)*\pi\left(q_k\right)\\ \sigma\left(q_j\right)&=\pi\left(q_i\right)*\sigma\left(q_k\right)+\pi\left(q_k\right)*\sigma\left(q_i\right)\\ \rho_j\left(q_j\right)&=\rho_i\left(q_i\right)+\rho_k\left(q_k\right)\\ \kappa\left(q_j\right)&=\kappa\left(q_i\right)+\kappa\left(q_k\right)\end{aligned} \quad (21)$$

$$\begin{aligned}&m^{-1}\left(e_j\right)*\left(1-\kappa\left(q_j\right)\right)*q_j\leqslant\\ &m^{-1}\left(e_i\right)*\left(1-\kappa\left(q_i\right)\right)*q_i+\\ &m^{-1}\left(e_k\right)*\left(1-\kappa\left(q_k\right)\right)*q_k\end{aligned} \quad (22)$$

$$\begin{aligned}&\left(m^{-1}\left(e_i\right)*\left(1-\kappa\left(q_i\right)\right)-\kappa\left(q_i\right)\right)*q_i+\\ &\left(m^{-1}\left(e_k\right)*\left(1-\kappa\left(q_k\right)\right)-\kappa\left(q_k\right)\right)*q_k\leqslant\\ &m^{-1}\left(e_j*\left(1-\kappa\left(q_j\right)\right)*q_j\right)\end{aligned} \quad (23)$$

If two series of questions (q_i and q_k) have identical questions then joint series has q_j questions which satisfy following expressions

$$\begin{aligned}q_j&=q_i=q_k\\ \pi\left(q_j\right)&=\pi\left(q_i\right)=\pi\left(q_k\right)\\ \sigma\left(q_j\right)&=\sigma\left(q_k\right)=\sigma\left(q_i\right)\\ \rho_j\left(q_j\right)&=\min\left(\{\rho_i\left(q_i\right)\}\cup\{\rho_k\left(q_k\right)\}\right)\\ \kappa\left(q_j\right)&=\kappa\left(q_i\right)=\kappa\left(q_k\right)\end{aligned} \quad (24)$$

Score e_j of the joint series is expressed

$$e_j=\min\left(\{e_i\}\cup\{e_k\}\right) \quad (25)$$

If two series of questions (q_i and q_k) have q_c identical questions then joint series has q_j questions which satisfy following

$$q_c\leqslant\min\left(\{q_i\}\cup\{q_k\}\right) \quad (26)$$

$$\begin{aligned}&2*k_c-q_c\geqslant\\ &\max\left(\{2*q_i*\kappa\left(q_i\right)-q_i\}\cup\{2*q_k*\kappa\left(q_k\right)-q_k\}\right)\end{aligned} \quad (27)$$

$$k_c\leqslant\min\left(\{q_i*\kappa\left(q_i\right)\}\cup\{q_k*\kappa\left(q_k\right)\}\right) \quad (28)$$

$$k_{ic}=q_i*\kappa\left(q_i\right)-k_c \quad (29)$$

$$k_{kc}=q_k*\kappa\left(q_k\right)-k_c \quad (30)$$

Score of the joint series is

$$m\left(\frac{\max\left(\{0\}\cup\{t_c-k_c-k_{ic}-k_{kc}\}\right)}{q_i+q_k-q_c-k_c-k_{ic}-k_{kc}}\right) \quad (31)$$

The task of navigation on tutorial materials is a part of navigation problem on hypermedia semantic networks [40], [42], [43]. Cooperative agents used solve it relies on the search query language. Kinds of agents depend on complexity of queries. The basic navigation interface operates with elementary queries [43].

Figure 2. System UI screenshots.

VI. Conclusion

The machines of integration platform were implemented using WebSockets with TCP communication protocol, browser javascript virtual machines and C\C++ compiler for Windows platform. The application was executed in mixed global and local area computer network environment. The server running ontology virtual machines has configuration including AMD Ryzen Threadripper 2950X processor with 8MiB of RAM and Windows 10 operating system. While configuration of clients has Windows 10 Intel Core i3 6100\i5 2310\2500 4\8MiB platform with Chrome, Firefox or Microsoft Edge browsers executing variety virtual machine UI implementation [38]. The described application was used during one semester period in purposes of help information and students knowledge testing system for two disciplines.

The future and perspective plans dealing with the integration platform includes: optimization and development of implementation of models of integration platform architecture to increase its performance and security qualities; implementation of new application of integration platform.

References

[1] William Clinger. Foundations of Actor Semantics. Mathematics Doctoral Dissertation. MIT, June 1981.

[2] Irene Greif. Semantics of Communicating Parallel Processes. EECS Doctoral Dissertation. MIT, August 1975.

[3] Gul Agha, Ian Mason, Scott Smith, Carolyn Talcott. A Foundation for Actor Computation. Journal of Functional Programming, January 1993.

[4] Carl Hewitt. What is Commitment? Physical, Organizational, and Social. April 2006.

[5] Carl Hewitt, Peter Bishop, Richard Steiger. A Universal Modular Actor Formalism for Artificial Intelligence. IJCAI, 1973.

[6] M. Brenner. A Multiagent Planning Language. Proceedings of the Workshop on PDDL. 13th International Conference on Automated Planning and Scheduling (ICAPS-200,3), Trento, Italy, 2003.

[7] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins. PDDL—The Planning Domain Definition Language. Technical Report CVC TR98003/DCS TR1165. New Haven, CT, Yale Center for Computational Vision and Control, 1998.

[8] D. L. Kovacs. A Multi-Agent Extension of PDDL3.1. Proceedings of the 3rd Workshop on the International Planning Competition (IPC). 22nd International Conference on Automated Planning and Scheduling (ICAPS–2012), Atibaia, São Paulo, Brazil, 2012. pp. 19–27.

[9] Manuela Veloso. PDDL by Example. Carnegie Mellon University, 2015.

[10] C.A.R. Hoare. Communicating Sequential Processes. Communications of the ACM, August 1978.

[11] S.D. Brookes, C.A.R. Hoare and W. Roscoe. A theory of communicating sequential processes, JACM, 1984.

[12] Robin Milner, A Calculus of Communicating Systems. Springer Verlag, 1980.

[13] Gul Agha. Actors: A Model of Concurrent Computation in Distributed Systems. Doctoral Dissertation. MIT Press, 1986.

[14] J.C.M. Baeten, T. Basten, M.A. Reniers. Algebra of Communicating Processes. Cambridge University Press, 2005.

[15] Robin Milner, Communicating and Mobile Systems: The Pi Calculus, Cambridge University Press, 1999.

[16] P.H.J. van Eijk et al., The Formal Description Technique LOTOS. North-Holland, 1989.

[17] ISO/IEC international standard 8807:1989. Information Processing Systems – Open Systems Interconnection – LOTOS: A Formal Description Technique based on the Temporal Ordering of Observational Behaviour. Geneva, September 1989.

[18] ISO/IEC international standard 15437:2001. Information technology – Enhancements to LOTOS (E -LOTOS). Geneva, September 2001.

[19] L. Cardelli, A.D. Gordon. Mobile Ambients. Proceedings of the First international Conference on Foundations of Software Science and Computation Structure (March 28 - April 4, 1998). M. Nivat, Ed. Lecture Notes in Computer Science, Springer-Verlag, 1998, vol. 1378, pp. 140–155.

[20] Shahram Rahimi. ACVisualizer: A Visualization Tool for Api-Calculus. October 2015.

[21] Jane Hillston. A Compositional Approach to Performance Modelling. Cambridge University Press, 1996.
[22] Cedric Fournet, Georges Gonthier. The Join Calculus: A Language for Distributed Mobile Programming. 2000.
[23] Robin Milner. Communication and Concurrency. USA, Prentice-Hall, 1989.
[24] Wil van der Aalst. Process Mining: Data Science in Action. 2016.
[25] W. M. P. van der Aalst, A. J. M. M. Weijters, and L. Maruster. Workflow Mining: Discovering process models from event logs, IEEE Transactions on Knowledge and Data Engineering, 2004, vol. 16.
[26] A. K. de Medeiros, W. M. P. van der Aalst, A. J. M. M. Weijters. Workflow Mining: Current Status and Future Directions. Lecture Notes in Computer Science, Springer-Verlag, 2003, vol. 2888.
[27] Davide Sangiorgi. Introduction to Bisimulation and Coinduction. Cambridge University Press, 2011.
[28] Robert M. Keller. Formal Verification of Parallel Programs. Communications of the ACM, July 1976, vol. 19, no 7, pp. 371—384.
[29] James S. Albus. 4D/RCS A Reference Model Architecture for Intelligent Unmanned Ground Vehicles. Proceedings of the SPIE 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Orlando, FL, 1-5 April 2002.
[30] John E. Laird. The Soar Cognitive Architecture. MIT Press, 2012.
[31] Yuwei Cui, Subutai Ahmad, Jeff Hawkins. "The HTM Spatial Pooler—A Neocortical Algorithm for Online Sparse Distributed Coding". Frontiers in Computational Neuroscience, 2017, Vol. 111, no 11.
[32] R. Orfali, D. Harkey. Java i CORBA v prilozheniyakh klient-server [Client/Server Programming with Java and CORBA] (Transl: Client/Server Programming with Java and CORBA, New York, 1998). Moscow, LORI, 2000, 712 p.
[33] B. Segall, D. Arnold, J. Boot, M. Henderson, T. Phelps. Content Based Routing with Elvin4. Proceedings AUUG2K, Canberra, Australia, June 2000.
[34] Psyclone – Communicative Machines Available at: https://cmlabs.com/psyclone (accessed 2021, May 5)
[35] Open Service Interface Definitions Available at: http://osid.org/ (accessed 2021, May 5)
[36] The Open Agent Architecture™. Available at: http://www.ai.sri.com/∼oaa/oaaslides/ (accessed 2021, May 5)
[37] version / openjsVVM / wiki / Home — Bitbucket (Variety Desert project). Available at: https://bitbucket.org/version/openjsvvm/wiki/Home (accessed 2021, May 5)
[38] Virtual Variety Machine. Available at: http://scnedu.sf.net/variety/_/index.html?variety=alterface.json&gradient=interscreen.json (accessed 2021, May 5)
[39] R. Solomonoff. Preliminary Report on a General Theory of Inductive Inference. Report V-131 (revision of the Feb. 4, 1960 report). Zator Co., Cambridge, Ma, November 1960.
[40] V.V. Golenkov, N.A. Gulyakina. Primenenie tekhnologii iskusstvennogo intellekta v obuchenii [Application of artificial intelligence technology in education] IV chteniya, posvyashchennye 70-letiyu so dnya rozhdeniya professora V.A. Karpova, 2010, pp. 14–16.
[41] V.V. Golenkov. Otkrytyi proekt, napravlennyi na sozdanie tekhnologii komponentnogo proektirovaniya intellektual'nykh sistem [An open project aimed at creating a technology for the component design of intelligent systems] Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems], 2013, pp. 55—78.
[42] V.P. Ivashenko. Modeli resheniya zadach v intellektual'nykh sistemakh. V 2 ch. Ch. 1 : Formal'nye modeli obrabotki informatsii i parallel'nye modeli resheniya zadach : ucheb.-metod. posobie [Models for solving problems in intelligent systems. In 2 parts, Part 1: Formal models of information processing and parallel models for solving problems: a tutorial] MInsk, BGUIR, 2020, 79 p.
[43] V.P. Ivashenko. Yazyk opisaniya sintaksicheskikh pravil dlya odnorodnykh semanticheskikh setei [Syntax rules description language for homogeneous semantic networks]. Distantsionnoe obuchenie – obrazovatel'naya sreda XXI veka, 2007. pp. 185–188.
[44] V.P. Ivashenko. Predstavlenie neironnykh setei i sistem produktsii v odnorodnykh semanticheskikh setyakh [Representation of neural networks and production systems in homogeneous semantic networks]. Izvestiya belorusskoi inzhenernoi akademii, 2003, vol. 1 no 1. pp. 184–188.
[45] V.P. Ivashenko. Spravochno-proveryayushchaya sistema na osnove unifitsirovannogo semanticheskogo predstavleniya znanii [Reference and testing system based on the unified semantic representation of knowledge] Informatsionnye tekhnologii i sistemy 2020 (ITS 2020) [Information Technologies and Systems 2020 (ITS 2020)], 2020, pp. 80—81.
[46] L.A. Kalinichenko. Metody i sredstva integratsii neodnorodnykh baz dannykh [Methods and tools for integration of heterogeneous databases]. Moscow, Nauka, Fizmatlit, 1983, 424 p.
[47] L.A. Kalinichenko, S.A. Stupnikov, V.N. Zakharov, Extending information integration technologies for problem solving over heterogeneous information resources, Inform. Primen., vol. 1, no 6, 2012, pp. 70—77.
[48] A.S. Narinyani. NE-faktory: netochnost' i nedoopredelennost' – razlichie i vzaimosvyaz' [Non-factors: inaccuracy and underdetermination – difference and interrelation]. Izv RAN (RAS). Ser. Teoriya i sistemy upravleniya 5 (2000). pp. 44—56.

Применение интеграционной платформы для решения задач, основанном на онтологических моделях, использующих унифицированное семантическое представление знаний

Ивашенко В.П.

В статье рассматривается решение в виде интеллектуальной интеграционной платформы, основанной на модели унифицированного семантического представления знаний, для разработки многоагентных систем, управляемых знаниями. В работе применяются: модель унифицированного семантического представления знаний на основе семантических сетей, модели и методы теории меры и теории вероятностей, методы дискретной оптимизации и прикладной математики, компьютерное моделирование и многоагентный подход. Работа направлена на разработку компьютерных средств с когнитивной архитектурой, использующих элементы искусственного сознания, способствующие гибкому взаимодействовию и адаптации этих средств в сложных образовательных приложениях. Были разработаны и реализованы виртуальные машины, другие подсистемы интеграционной платформы, а также - справочно-проверяющая прикладная многоагентная система, функционирующие в рамках системы человеко-машинного взаимодействия.

Received 14.05.2021