J Log Lang Inf (2014) 23:331–345
DOI 10.1007/s10849-014-9195-y

Functional and Structural Integration without Competence Overstepping in Structured Semantic Knowledge Base System

Marek Krótkiewicz · Krystian Wojtkiewicz

Published online: 4 April 2014

Abstract Logic, language and information integration is one of areas broadly explored nowadays and at the same time promising. Authors use that approach in their 8 years long research into Structured Semantic Knowledge Base System. The aim of this paper is to present authors idea of system capable of generating synergy effect while storing various type of information. The key assumption, which has been adopted, is the thesis that the attempt to find universal way of the reality description is very inefficient solution. Combination of several solutions into one system must be based on the principle of supporting rather than on redundancy, ambiguity or mutual antagonism areas of logic, language and information. Natural language processing in the context of logics or information processing is ineffective, not to say pointless in the means of certainty. It is simply caused by the fact that natural languages have been formed long time ago as the main communication channel, rather than for automatic information processing. It was one of the reasons that logic has been founded as the universal language of mathematics and formalized communication in more extensive sense. Information processing however brought more sophisticated problems that mathematics could not smoothly solve, what made computer science appear. Studies over artificial intelligence revealed even more complex issues to solve. The unity of those three areas: logics, language and information is a necessity to acquire complementarity and synergy.

Keywords Artificial intelligence · Semantic network · Ontological core · Computer linguistics · Knowledge base · Semantic knowledge base · Information processing

M. Krótkiewicz
Institute of Control and Computer Engineering, Opole University of Technology, Opole, Poland

K. Wojtkiewicz (✉)
Department of Biosystems Engineering, Opole University of Technology, Opole, Poland
e-mail: krystian.wojtkiewicz@gmail.com

1 Introduction

Concepts described by terms *information, logic* and *language* are strongly bound together, however these relations are not simple like generalization–specialization, part-whole or any other named relations. *Information* is used as description of part of reality. This description is build using concepts known and being used by the agent who is the owner of that information. There are many ways of describing reality and they depend heavily on the internal structure and mechanisms available to the agent. In the case of human agent information is stored in the brain in the form of neural connections and in fact it is not clear what is the physical representation of the mechanism. In the case of computer systems, information is stored in categories defined in its model, for example, a way to store information are the databases. Each database is defined in a model, such as hierarchical, network model, relational, object-oriented, graph-oriented or association-oriented (Liskov 1988; Minoura and Iyengar 1989; Liu and Zhou 2008; Zhou et al. 2010). Each of models defines set of categories and mechanisms underlying information storage.

Language is one of the communication channels that makes transfer of information from one agent to another possible. Therefore primary, and some say that the only, function of language is to allow the transfer of information from source to destination. One of the most important issues is to perform this task in a way that does not disrupt the information being transmitted. Information can be properly interpreted only if compliance does not occur only in grammar, on both sides of the communication channel, but also in terms of semantics. Semantics here is pointing at meaning of each of elements building the information.

Logic is the narrowest way, can be used as description of rules for information processing, e.g. inference. One of the functions of logic is to deliver functionality of language, however the assumption made by the authors is that it should carry out its tasks apart from the transmission (language) channel. One of the main problems is proper transformation of information sent by the language into agent internal logical rules. One solution is to transfer information using language of logic. However, given that the agents involved in the interaction of communication can and often are the people, this approach is very inefficient and difficult for practical implementation. In turn, the use of natural language makes the issue of information translated into language of logic quite difficult. Just as there are no general rules for natural or quasi-natural language processing (NLP) methods to the form of information stored in databases, there is no simple way of using the information contained in those databases during process of grammatical rules formation.

2 Semantic Knowledge Base

Semantic Knowledge Base (*SKB*) (Krótkiewicz and Wojtkiewicz 2005, 2009, 2013a,b) is a system designated to store information. It is designed in a way to ensure these two previously mentioned interfaces, namely information-language and information-logic. Although the ability to store information is very important, *SKB* would be just another database, if it did not have the ability to convert information into a quasi-

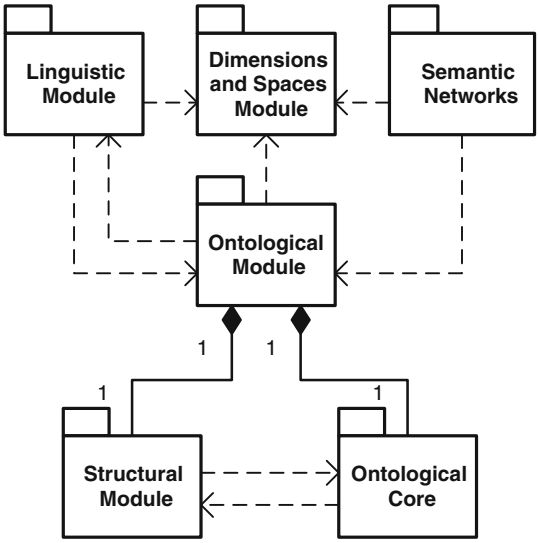

Fig. 1 Semantic Knowledge Base modules

natural language and store information in the form of logical rules. Information may have very different nature, hence the *SKB* has several specialized modules (see Fig. 1) that enable the integration of the specifics. What is even more important, *SKB* has specialized modules providing the ability to transform and store information in the form of logical rules, what is mainly derived from Semantic Network Module and Dimensions and Spaces Module.

2.1 Knowledge Base Modeling Approach

SKB was designed with the use of Association-Oriented Database (AODB) model. Its due to the fact this model (AODB) has been developed especially for the purpose of knowledge management and processing systems. At the preliminary stage of SKB development it has been discuss whether not to use object-oriented model along with the standardized modelling language UML. This idea has been abandoned and the most important reasons are provided in following chapter.

UML is conceptual language since it is detached form specific implementation. It is its great advantage but at the same time this feature is associated with certain danger. UML refers to the object model, which in turn has different implementations in the form of programming languages. These languages, despite operating in the area of object-oriented programming, significantly vary even in such basic issues as inheritance. Especially the approach to multiple inheritance. In term of databases, the case is even more difficult, because as of today all attempts towards one unified object-oriented database model standard failed. ODMG 3.0 still remains just a reference rather than any official standard since it is based on too many compromises that make it far unimplementable. Consequently, there are many object-oriented database models that willingly relate to ODMG 3.0 as to approach that has to be less or more modified.

UML is a modeling language and as such it is standing above implementation languages. Therefore, the diagrams can be implemented differently depending on the applied language or OODB in case of databases. The use of UML as the object-oriented modeling language is very convenient and efficient as it provides the ability to share information with huge number of scientists, engineers and designers. Note, however that the language is not perfect, it has traps that for people who are not familiar with them might cause problems. Particular emphasis should be placed on the diagram-implementation transition. It is ambiguous and may cause errors and inconsistency for the modeled systems.

It was one of the reasons leading to the decision that the structure of the SKB will not be expressed in UML. This decision has caused another one, that SKB shall not be expressed in UML, but in general the object-oriented approach will be left behind. The System has been developed using AODB model and its modeling language—Association-Oriented Modeling Language (AML). AML is a new graphical based modeling language therefor it is strictly bound with formally defined association-oriented data definition language. AML is defined strictly for the usage of Association-Oriented Model. The aim of AML is to provide similar function to class diagram in UML, however it is structure oriented diagram, so it has categories that stick to structural categories of AODB e.g. class, association or role. AML grammar and semantics is different from the UML class diagram in terms of categories and therefor it would be hard to map either of diagrams in the models they were not defined for. Unlike in Object-Oriented approach the language has been developed along with application and modeling methodology. AML as part of Association-Oriented Model will be the subject of separate paper, as well as a separate section in book being currently prepared for print (Krótkiewicz Marek 2014).

2.2 Ontological Module

The core of *SKB* is formed by *Ontological Module*, which consist of *Ontological Core* and *Structural Module*. The task given to *Ontological Core* is to store the description of concepts, through their properties and simple relationships e.g. *class-instance*, *instance-set*, *feature-value*, etc. *Concepts* in *SKB* hold the meaning of terms, They are being defined through basic attributes derived from collection assignment and, what is more important, through sets of links with other concepts. Concept itself is abstract, what is understood here as no direct connection to any name in any natural language. Terms describing concepts can be found in *Linguistic Module* (Krótkiewicz and Wojtkiewicz 2013a) and the relations between modules provide the ability to refer to concepts by names in any language.

2.3 Structural Module

Structural Module can store more complex information than the *Ontological Core Module*. Using this module concepts can be combined into relations, either predefined in the system, such as *generalization–specialization*, *whole-part*, etc., as well as any user-defined relation. By analogy with object-oriented modeling, structural module

is a meta-structure adequate to class diagram and objects diagram at the same time. *Structural Module* operates only on concepts previously defined in the *Ontological Core*.

2.4 Semantic Network Module

Semantic Network Module is the most complex and most versatile module in term of information storage. It uses structures based on semantic networks e.g. nodes and edges, but significantly enhanced in idea compared to traditional semantic network approach. In particular, multiplicity of relations, quantifiers, confidence coefficient modifiers has been added. Semantic networks in *SKB* are based on the structure of the *operator-operand*. This means that the basic structural unit—*operator* specifies how operands are tied together.

Both operands and operators are subject to **modification**, i.e. some details can be added regarding description of particular semantic network element, e.g.:

Mary likes ice cream. (Simple network)

$$likes\left\{\overset{Actor}{\longleftrightarrow}\langle Mary\rangle, \overset{Object}{\longleftrightarrow}\langle icecream\rangle\right\} \tag{1}$$

Fat Mary likes good ice cream. (Network with modifiers)

$$likes\left\{\overset{Actor}{\longleftrightarrow}\langle Mary \overset{\cdot}{\cdot} fat\rangle, \overset{Object}{\longleftrightarrow}\langle icecream \overset{\cdot}{\cdot} good\rangle\right\} \tag{2}$$

Very fat Mary likes quite good ice cream. (Network with modified modifiers).

$$likes\left\{\begin{array}{l}\overset{Actor}{\longleftrightarrow}\langle Mary \overset{\cdot}{\cdot} fat \overset{\cdot}{\cdot} very\rangle,\\ \overset{Object}{\longleftrightarrow}\langle icecream \overset{\cdot}{\cdot} good \overset{\cdot}{\cdot} quite\rangle\end{array}\right\} \tag{3}$$

Quantifiers basically are used to describe specifics of time and space. They are related to such terms in natural language as: *never, always, anywhere, somewhere*.

Multiplicity is used in situations, in which undefined relation multiplicity create serious interpretation problem, e.g.:

The dog has a paw.

$$have\left\{\overset{Actor}{\longleftrightarrow}\langle dog\rangle, \overset{Object}{\longleftrightarrow}\langle paw\rangle\right\} \tag{4}$$

Everyone understands that a dog has four paws, not one. It is just agent interpretation, therefore this sentence is not clear for system and has to be supplemented with multiplicity.

*The dog has **four** paws*.

$$have\left\{\overset{Actor}{\longleftrightarrow}\langle dog\rangle, \overset{Object}{\longleftrightarrow}\langle paw[4]\rangle\right\} \tag{5}$$

To make it even more precise, and provide clearly defined relationship multiplicity, one should say:

*The dog has **four** paws and each paw is assigned only to one dog.*

$$have \left\{ \overset{Actor}{\longleftrightarrow} \langle dog[1] \rangle, \overset{Object}{\longleftrightarrow} \langle paw[4] \rangle \right\} \tag{6}$$

Certainty factor (CF) may be assigned to each of the elements of sentence (network). This means that stating any statement, one cannot only assign a probability factor to sentence itself, but also to determine the probability of each of its items, e.g.:

Very fat Mary (95 % confidence), like quite good ice cream (probability of 5 %).

$$likes \left\{ \begin{array}{l} \overset{Actor}{\longleftrightarrow} \langle Mary^{0.95} \cdot_{\cdot} fat \cdot_{\cdot} very \rangle, \\ \overset{Object}{\longleftrightarrow} \langle icecream^{0.05} \cdot_{\cdot} good \cdot_{\cdot} quite \rangle \end{array} \right\} \tag{7}$$

In the given sentence, written in natural language, the author described that subject of the sentence (actor) is Mary with the confidence level of 95 % and the object is ice cream, wherein the probability of that it is ice cream is 5 %. The reference to the probability has been pointed in the aspect of information storage. At this point no issues regarding calculation of the probability of the sentence are considered. The essence of information storage within Semantic Networks module of *SKB* is the possibility of assigning of the CF to each and every element of the semantic network independently.

At this point, the calculation of the probability of the truth of a sentence is not so important, but you can easily adopt the principle of the product of the probabilities of components. We could assume probability of the occurrence of the dependent, but again calculating the probability of a sentence depends on the adopted algorithms and has no reflection on the structure, and as such is not part of this study. From the *SKB* structure point of view, it is possible to assign the probability for virtually every component of the structure, i.e. both the node and edge in the semantic network, regardless of their nature.

2.5 Linguistic Module

Linguistic Module (Krótkiewicz and Wojtkiewicz 2013a) is used to associate concepts and terms. It contains structures for natural language descriptions of words and lexemes. The module uses the structure module to describe any structural relationships between words. This is done completely analogously, as in the case of concepts, e.g. for the defined concept of a relationship, such as *generalization–specialization*, one can indicate that some word A is a special case of (derivative) word B. *Linguistic Module* was built to allow communication with *SKB* using a quasi-natural language. It is a kind of interface between information and communication layer. This module is not specialized for a particular language or group of languages, and its role is to manage the terms. Natural language grammar is outside the *SKB* structure.

2.6 Dimensions and Spaces Module

Dimensions and Spaces Module is another part of *SKB*, which main purpose is to allow assignment of location in specific point in any space to information stored in *SKB*. It is assumed that space is defined through dimensions. Each dimension and space has direct connection with concept defined in ontological core of *SKB*. It is possible to implement literally any kind of dependencies between dimensions forming spaces. Given the above, Kripke's possible world theory (Hazen 1982; Copeland 2002; Akinci 2003) might be implemented in this module, as well as any other idea regarding relativity of information. In the narrowest term, this module is being used to place facts in time and space and to determine the scope of rules in a given dimension. Database structure of the module puts minimal restraints on the possibility of dimension or space definition. The issue of semantics definition, as well as further processing has been moved into implementation (algorithms) layer of the system.

3 Information Storage

In order to present some of the mechanisms used by *SKB* to store information simple facts has been prepared. They will be written in natural language using terms understandable by human agents. System itself is working on concepts being abstract identifiers that using linguistic module can be identified by natural language terms. Each fact entered into the system may have its emanation in various forms of information stored in the system. These forms are due to the individual modules. *Semantic Networks Module* is sufficiently general structure that can save very wide variety of information types. In particular, it makes possible to store information that can and should be stored in other modules. This is due to the fact that the nature of the information should be closely matched to the structure of its representation, which greatly speeds up the process. To illustrate this issue authors will present the following example. The sentence in natural language: *Kate is a girl*. The term '*is'* is treated here as a *class-instance* dependency, which means that this phrase should be interpreted as *Kate is an instance of the class Girl*. In this point of view, the *"is"* operator is considered to be a binary operator *is(x,y)* with *x=Kate* and *y=girl.* On the other hand the very same sentence could be grammatically parted using *is-girl* as a unary operator. Author decided that for the clarity of the main track of the paper the first option of interpretation will be explored in this and any following examples. The issue of grammatical and semantic ambiguity is not in the main track of this paper, however plays a huge role in SKBs and will be directly addressed by authors in another publication regarding knowledge representation in SKB.

This sentence can also be represented in the semantic network where the operator would be *is*, while Kate and the girl would be operands. As previously mentioned, the information storage location should be tailored to the nature of the information. Therefore, the semantic network module is not optimal, since the *Ontological Core* is a specialized structure according to the *Class-Instance* scheme.

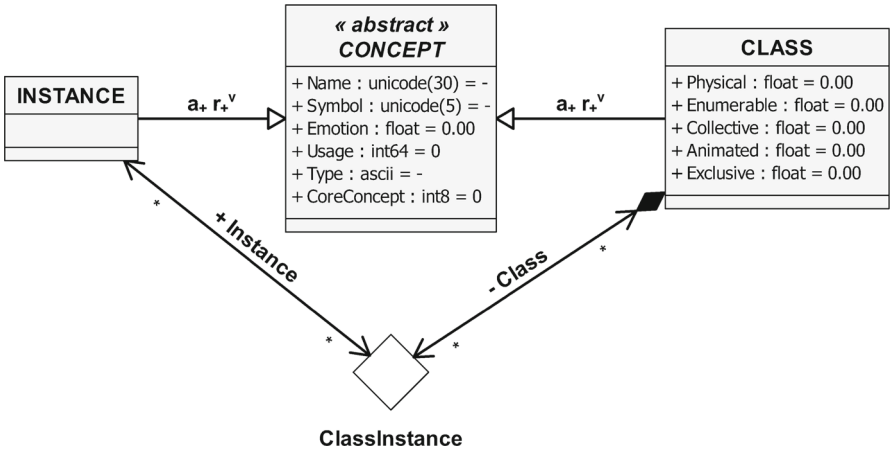

Fig. 2 Part of Ontological Core structure responsible for Class-Instance scheme building

The formal definition of the *Ontological Core* structure presented on Fig. 2 is the following:

$$ClassInstance\left\langle \begin{array}{l} [*]^{+Instance}\longleftrightarrow [*]\,INSTANCE, \\ [*]^{-Class}\longleftrightarrow \blacklozenge\,[*]\,CLASS \end{array} \right\rangle \tag{8}$$

$$INSTANCE -\triangleright^{a_+ r_+^v} CONCEPT^{\emptyset}; CLASS -\triangleright^{a_+ r_+^v} CONCEPT^{\emptyset} \tag{9}$$

With formal information storage scheme:

$$INSTANCE\langle o_1\rangle; CLASS\langle o_2\rangle \tag{10}$$

Which means that object o_1 is part of collection *INSTANCE*, while object o_2 is part of collection *CLASS*.

$$o_1\langle Kate : Name\rangle; o_2\langle girl : Name\rangle \tag{11}$$

For the record, this means that attribute *NAME* of object o_1 has value *Kate*, while the attribute *NAME* of object o_2 has value *girl*.

$$ao_1 : ClassInstance\left\{ \begin{array}{l} {\scriptstyle Instance:INSTANCE}\longleftrightarrow aor_1\langle o_1 : INSTANCE,\rangle \\ {\scriptstyle Class:CLASS}\longleftrightarrow aor_2\langle o_2 : CLASS\rangle \end{array} \right\} \tag{12}$$

The information would be stored using one *Association Object*: ao_1 that would bind objects o_1 and o_2. There would also be two *Association Object Role*: aor_1 and aor_2 that mainly are used to group objects into lists. In this case lists consist only of one element, but they are capable to store more objects, e.g.:

Fig. 3 An example of Class-Instance assignment in SKB

$$INSTANCE\langle o_1, o_2, o_3\rangle; CLASS\langle o_4\rangle \tag{13}$$

$$o_1\langle Rose : Name\rangle; o_2\langle Mary : Name\rangle; \tag{14}$$

$$o_3\langle Kate : Name\rangle; o_4\langle girl : Name\rangle \tag{15}$$

$$ao_1 : ClassInstance \left\{ \begin{array}{l} \overset{Instance:INSTANCE}{\longleftrightarrow} aor_1\langle o_1, o_2, o_3 : INSTANCE,\rangle \\ \overset{Class:CLASS}{\longleftrightarrow} aor_2\langle o_4 : CLASS\rangle \end{array} \right\} \tag{16}$$

These are formal definitions derived from *AODB Model*, in which *SKB* has been designed. There is also shorter version of formal definitions, i.e.:

$$ClassInstance\{Instance\langle Kate\rangle, Class\langle girl\rangle\} \tag{17}$$

as well as for the second example:

$$ClassInstance\{Instance\langle Rose, Mary, Kate\rangle, Class\langle girl\rangle\} \tag{18}$$

where *ClassInstance* is the name of *Association*, *Instance* and *Class* are names of roles of objects: *Rose*, *Mary*, *Kate* and *girl* respectively.

Simplified, graphical version of this information has been showed on the Fig. 3.

The sentence in natural language: *Tom likes women* is another information example. This information has to be written in the *Semantic Network Module* since *like* is not one of predefined associations build into any other module, in particular *Ontological Core*. Formal representation of this fact has the following form:

$$INSTANCE\langle o_1\rangle; RELATIONSHIP\langle o_2\rangle; \tag{19}$$

$$CLASS\langle o_3\rangle; ROLE\langle o_4\rangle; ROLE\langle o_5\rangle \tag{20}$$

$$o_1\langle Tom : Name\rangle; o_2\langle likes : Name\rangle; o_3\langle women : Name\rangle \tag{21}$$

$$o_4\langle Actor : Name\rangle; o_5\langle Object : Name\rangle \tag{22}$$

$$ao_1 : SubNode \left\{ \begin{array}{l} \overset{Operator:OPERATOR}{\longleftrightarrow} aor_1\langle o_2 : RELATIONSHIP\rangle, \\ \overset{Operand:OPERAND}{\longleftrightarrow} aor_2\langle o_1, o_3 : INSTANCE\rangle, \\ \overset{OperandRole:ROLE}{\longleftrightarrow} aor_3\langle o_4, o_5 : ROLE\rangle \end{array} \right\} \tag{23}$$

Objects o_4 and o_5 are predefined system roles. *Actor* means that defined in relationship (23) element can be treated as activity, state or it defines property. Contrary, *Object* means that this action, state or property is being used to describe *actor*. This mechanism is essential to clarify the relation in the means of its direction. The above

notation does not include all elements necessary to build a more complete semantic network for this exemplary information. The diagram below (see Fig. 4) shows a part of the *SKB* structure, which concerns *Semantic Networks Module*. For simplicity, it does not take into account the elements describing quantifiers. Quantifiers are a powerful mechanism, which will be an object of a separate study.

Information *Tom likes women* can be presented in *AODB Model* formal way like follows:

$$likes \left\{ \overset{Actor}{\longleftrightarrow} \langle Tom \rangle, \overset{Object}{\longleftrightarrow} \langle women \rangle \right\} \tag{24}$$

or

$$\langle Tom \rangle \overset{Actor}{\longleftrightarrow} likes \overset{Object}{\longleftrightarrow} \langle women \rangle \tag{25}$$

Presented sentence has an element *Tom*, which is easily identified by a human agent as an identifier of instance belonging to class *human*. However, from the system point of view, not working on the terms, there will be no text that could be interpreted in this way. Therefore, if the system does not have information about concept used, in this case about concept referred to by the term *Tom*, the first thing that it performs is to generate a request for clarification what a given concept is. The most important part of an answer to this request is to determine to which concept category it belongs. The *SKB* distinguish a number of predefined categories such as *Class*, *Instance*, *Feature*, etc. If it is determined that *Tom* is an *Instance*, the system must determine what *like* is. In this example, *like* is the relationship, while the *women* is a *Class*. The semantics of the information stored in the semantic network is as follows: *Instance Tom* is connected with the *Class* of *women* through relationship *like*. From the point of view of semantic networks, this information should be read as follows: the *Operator like* has two *Operands Tom* and *women*. This information must be accompanied by a statement which operand acts as an *actor*, and which is the *Object*. This information is stored in following structure:

$$SubNode \left\langle \begin{array}{l} [*] \overset{+Operator:OPERATOR}{\longleftrightarrow} \blacklozenge [1..1]\, OPERATOR, \\ [*] \overset{+OperandRole}{\longleftrightarrow} [1..1]\, ROLE, \\ [*] \overset{+Operand}{\longleftrightarrow} [*]\, OPERAND \end{array} \right\rangle \tag{26}$$

or to be precise:

$$SubNode\, [*] \overset{+OperandRole}{\longleftrightarrow} [1..1]\, ROLE \tag{27}$$

With the addition of information on the category to which operands belong, and what role they have in the sentence, stored fact has been somewhat clarified. Please note that this is not the end of *SKB* features regarding storage information in semantic networks. There are modifiers, quantifiers and a number of attributes associated with them, but the issue is complex and far exceeds the volume of this paper.

Exemplary *generalization-specialization* relations, which are stored in *Structural Module* of *SKB*, has been showed on Fig. 5.

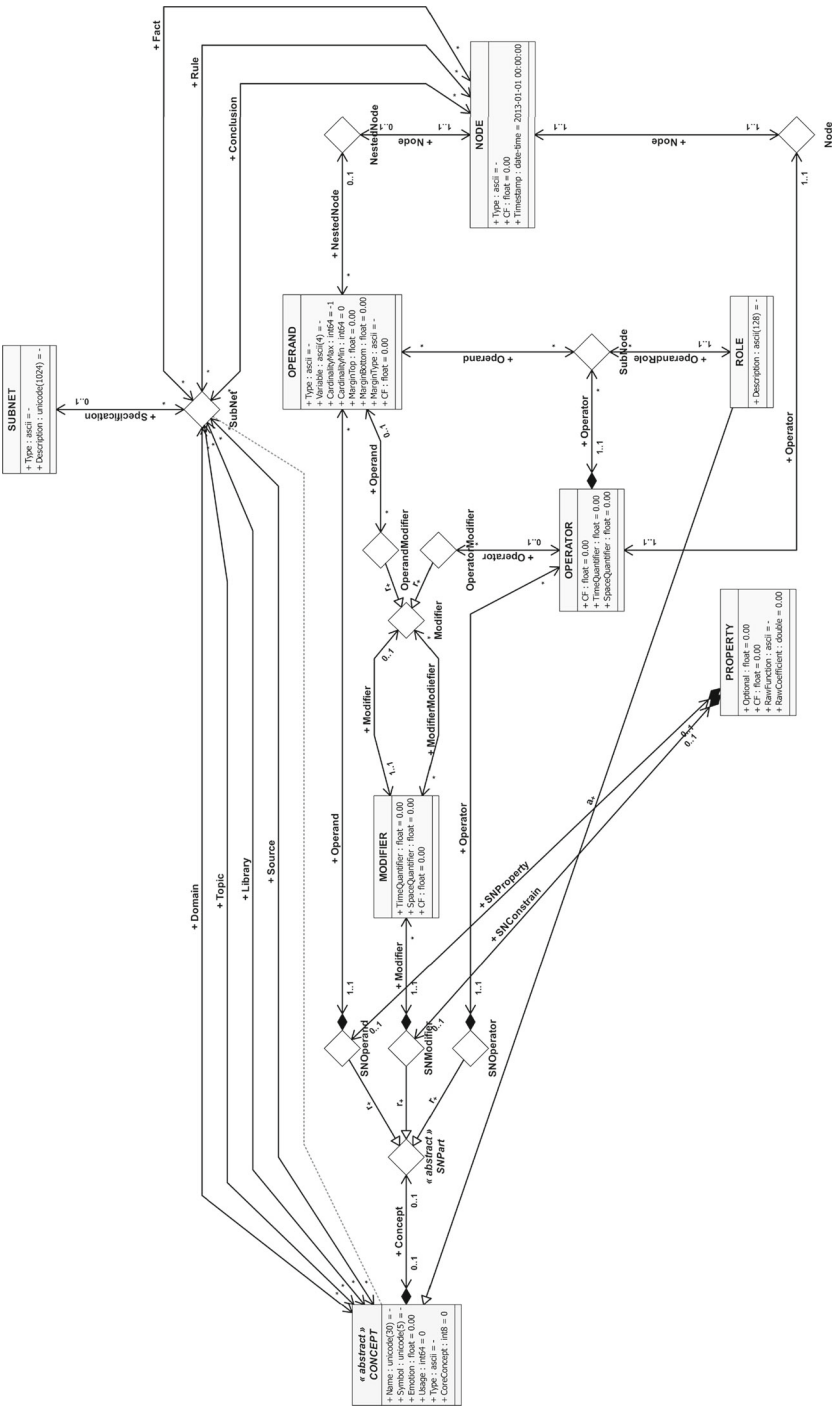

Fig. 4 Simplified Semantic Network module structure

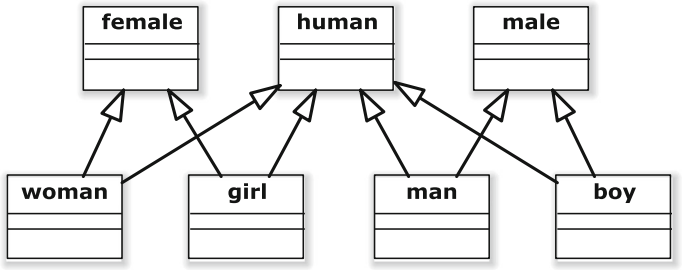

Fig. 5 An example of generalization–specialization relation

The *Structural Module* is built in such a way that it store information about the structural relationships in terms of UML[1] diagrams known as class diagrams and object diagrams (see Fig. 6).

This can be presented in formal definition according to *AODB Model* as follows:

$$LinkTypeRoles \left\langle \begin{array}{l} \cdots LINKDESCRIPTION, \\ [0..1] \overset{+Linktype}{\longleftrightarrow} \blacklozenge [1]\, LINKTYPE, \\ [0..1] \blacklozenge \overset{+Roles}{\longleftrightarrow} [*]\, ROLETYPE \end{array} \right\rangle \tag{28}$$

$$LinkRoleRoleType \left\langle \begin{array}{l} [0..1] \overset{+RoleType}{\longleftrightarrow} [1]\, ROLETYPE, \\ [1] \overset{+LinkRole}{\longleftrightarrow} [*]\, LINKROLE \end{array} \right\rangle \tag{29}$$

$$Link \left\langle \begin{array}{l} \cdots CONCEPT, \\ [*] \overset{+LinkType}{\longleftrightarrow} [1]\, LINKTYPE, \\ [1] \blacklozenge \overset{+LinkRoles}{\longleftrightarrow} [*]\, LINKROLE, \\ [*] \overset{+Aspect}{\longleftrightarrow} [*]\, CONCEPT \end{array} \right\rangle \tag{30}$$

$$LinkRoleConcept \left\langle \begin{array}{l} \cdots LINKDESCRIPTION, \\ [*] \overset{+LinkRole \cdots CONCEPT}{\longleftrightarrow} \blacklozenge [1]\, LINKROLE \end{array} \right\rangle \tag{31}$$

Structural Module consist of two conceptual elements. The first is a system used for building relation templates, defining type of relation and roles building it. It corresponds to UML class diagram. The second one is used to determine the specific relationships between concepts defined in *Ontological Core*. It corresponds to object diagram in UML. Very important issue is that in the process of defining relations and roles only concepts previously defined in *Ontological Core* may be used. Given that *Ontological Core* module is capable of defining any terms, which later might become components of relations in the *Object Module*, it should be noted that it gives ability to create any possible relations. Therefore this module is not limited to standard relations, e.g. *whole-part*, *generalization–specialization*.

[1] Unified Modelling Language

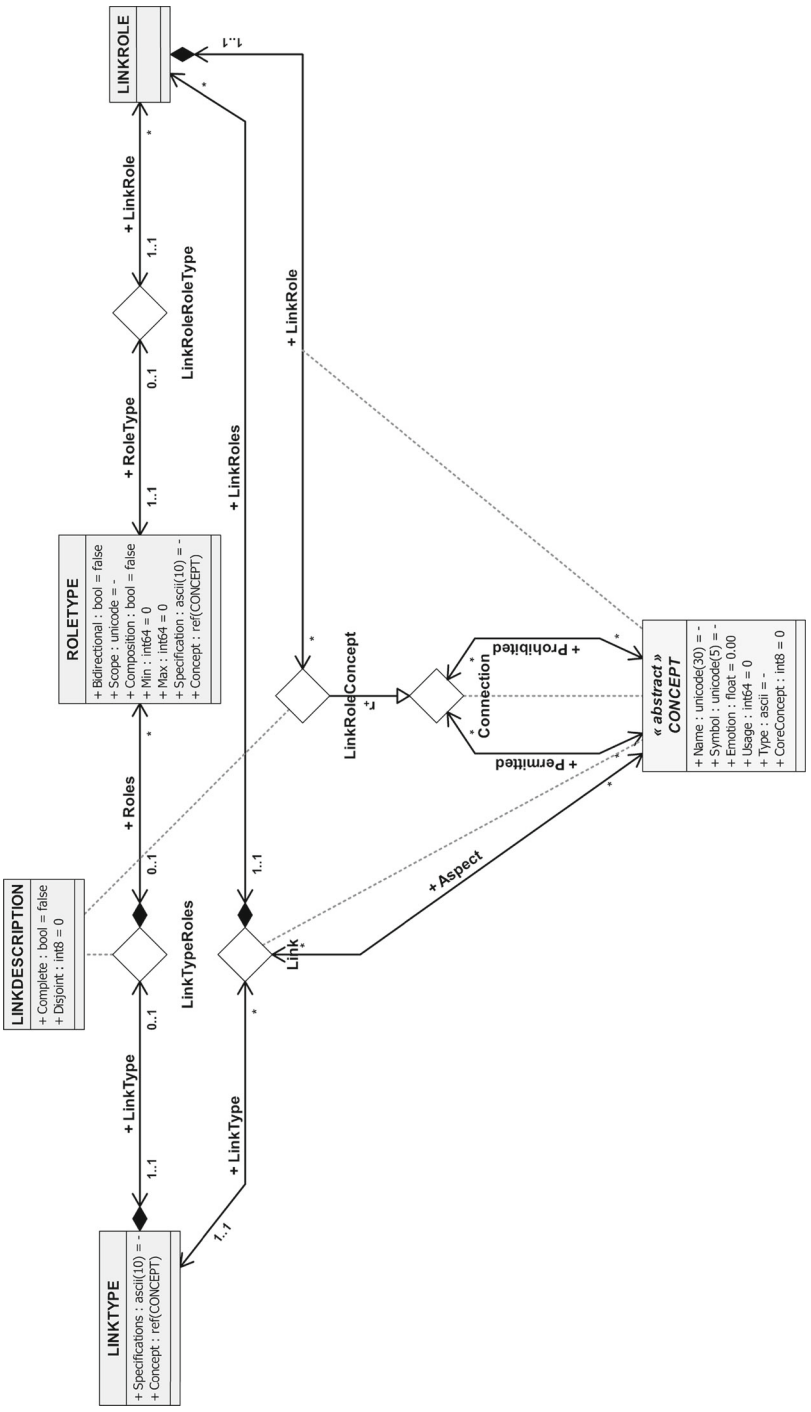

Fig. 6 SKB Structural Module diagram

4 Summary

Presented approach requires to take into consideration natural, structural and functional limitations of elements: logic, language and information. As natural languages should not be used to information processing or its storage, logics will never become efficient in human-system communication. Each element has its own fields of implementation and the problem is in building adequate unity. The system presented in this paper is an original solution build from scratch independently of existing solutions. Therefore it is not continuation of the work on any of the existing systems. This applies to each of the SKB layers, including the lowest layer, i.e. database. However, authors reviewed the existing solutions and systems in the attempt to capture the most important and most valuable ideas but not limited to specific applications. The conclusion of the review was to develop a hybrid system build from elements such as semantic networks, ontology, rule or logic-based systems as well as solutions derived from the NLP field. The solution that has been built does not constitute just simple aggregation of those ideas. It is rather a way to combine these concepts into a coherent system of mutually reinforcing elements. The main weakness of systems based on natural languages is automated processing. They also have almost no effective approach to interpretation of the terms or expressions. Logic based systems have strong inference methods but on the other side provide minimal support towards translations of facts or rules into any language that could be easily understood by human being. There are also ontology based systems, which in turn are primarily focused on elementary relations between classes and objects, but in most do not provide any solution in storing complex facts and rules. Contrary, semantic network provide mechanism for complex facts storage while they prove inefficient with simple relation.

The solution proposed by the authors has three layers of information storage, namely: conceptual, information and language layers. Each of them has its own functionalities. Some of those functionalities are internal operations, whereas others are interfaces between layers. Taking other points of view, the structure of the *SKB* is quite complex. There are specialized modules: *Ontological Module* (*Ontological Core* + *Structural Module*), *Behavioral Module*, *Dimensions and Spaces Module*, *Linguistic Module*, *Semantic Network Module*. The last listed module is the most important in the aspect of logics implementation and formalized information processing. This module is based on, extended by authors, semantic network idea using roles, quantifiers, multiplicity, certainty, error margin over the structure of operators and operands. Taking the modules and the layers described, authors created the system where each of the fields represented by logic, language and information is strictly imposed in the structure and functionalities.

Logic, language and information are complementary and as such to support each other they overlap and link to each other. Those links and overlaps got realized in the form of *SKB*. The system generates synergy through the fact that once information is stored in one module it becomes complementary to the others. This creates added value as well as the fact that information is not replicated in various forms. It is not that easy to achieve such an effect by simple combination of separate systems working on logic, language and information. One of the things that has to be taken into consideration is how to deal with complimentary information stored in more than one module. There

are several strategies that can be enforced to sustain intra-consistency of the system, therefor thoughtful and accurate determination of both structure of individual modules and their relation, as presented in this paper, is needed.

Finally, what has to be noted, this effect would not be possible if not for common database structure definition layer. For the *SKB* this layer has been *AODB Model.*

References

Akinci, S. (2003). A classification of the approaches to the ontology of possible worlds. In A.-T. Tymaniecka (Ed.), *Does world exist plurisignificant ciphering real* (pp. 855–866). Berlin, Heidelberg: Springer Netherlands.

Copeland, B. J. (2002). The genesis of possible worlds semantics. *Journal of Philosophical logic*, *31*, 99–137. doi:10.1023/A:1015273407895.

Hazen, A. (1982). On a possible misinterpretation of Kripke's semantics for intuitionistic logic. *Analysis*, *42*, 128–133. doi:10.1093/analys/42.3.128.

Krótkiewicz, M., & Wojtkiewicz, K. (2005). Conceptual ontological object knowledge base and language. In M. Kurzyński, E. Puchała, M. Woźniak & A. Żołnierek (Eds.), *Comput. Recognit. Syst.* (pp. 227–234). Berlin, Heidelberg: Springer.

Krótkiewicz, M., & Wojtkiewicz, K. (2009). Knowledge acquisition in conceptual ontological artificial intelligence system. In Z. S. Hippe & J. L. Kulikowski (Eds.), *Human-computer systems interaction* (pp. 29–37). Berlin, Heidelberg: Springer.

Krótkiewicz, M., & Wojtkiewicz, K. (2013a). *Introduction to semantic knowledge base*. Linguistic Module.

Krótkiewicz, M., & Wojtkiewicz, K. (2013b). An introduction to ontology based structured knowledge base system: Knowledge acquisition module. In A. Selamat, N. T. Nguyenl, & H. Haron (Eds.), *Intelligent information and database systems* (pp. 497–506). Berlin, Heidelberg: Springer.

Krótkiewicz, M. (2014). *Introduction to association oriented model*. (to be published).

Liskov, B. (1988). Keynote address-data abstraction and hierarchy. *SIGPLAN Notices*, *23*, 17–34. doi:10.1145/62139.62141.

Liu, X. L. X., & Zhou, B. Z. B. (2008). Research on knowledge base of dam safety monitoring expert system based on relational database. *IEEE International Symposium on Acquisition and Modeling Workshop, 2008* (pp. 0–3). doi:10.1109/KAMW.2008.4810689.

Minoura, T., & Iyengar, S. S. (1989). Data and time abstraction techniques for analyzing multilevel concurrent systems. *IEEE Transactions on Software Engineering*, *15*, 47–59. doi:10.1109/32.21725.

Zhou, S., Meng, G., & Ling, H. (2010). Ontologies acquisition from relational databases. *Computer & Information Science*, *3*, 185–187.