Ontology-based Design of Intelligent Systems

Golenkov V.V.
Belarusian State University of Informatics and Radioelectronics
Minsk, Belarus
golen@bsuir.by

***Abstract*—The main subject of this paper is the consideration of technology of designing intelligent systems, based on the *ontologies*, i.e. ontological treatment of this technology. Ontological approach to design of any complex system classes (including intelligent systems) makes it possible to decompose clearly and hierarchically a process of *designing* any system of specified class into such *design actions*, many of which can be executed in parallel and each of which can be executed locally, i.e. within the specific ontologies. Within the paper basic principles of such an approach are given.**

***Keywords*—ontology, subject domain, ontology-based design, intelligent system, Ontology of intelligent systems, Ontology of designing intelligent systems, language for sense knowledge representation, formal ontological model of intelligent system.**

I. Introduction

A. Objective and Relevance of the Work

The purpose of this paper is to make more precise definition for concept of ***formal model of intelligent system***, and also to consider methodology and tools for ***designing intelligent systems***. The aggregate of such models, methodologies and tools is nothing but ***the technology of designing intelligent systems***. Relevance of creating such technologies is due to expansion of ***intelligent systems*** applying area and, as a consequence, due to the necessity to decrease essentially the work content for its development.

The feature of this paper is consideration of technology of designing intelligent systems on the base of ***ontologies***, i.e. ontological treatment of this technology [6], [7], [27], [28], [22], [19], [20]. Ontological approach to designing any complex system classes (including intelligent systems) makes it possible to decompose clear and hierarchically a process of ***designing*** any system of specified class into such ***design actions*** many of which can be executed in parallel and each of which can be executed locally, i.e. within the framework of specific ontologies.

Thus hierarchical system of design actions is put in correspondence with hierarchical structure of ontology within a framework of which such design actions can be carried out. It is clear that such an approach speeds up essentially design activity by means of multisequencing and localization of an area of searching a solution while executing each design action.

B. Problems to be Solved to Succeed

- To improve effectiveness of designing intelligent systems it is necessary to have a ***general*** (complex, integrated, holistic) ***technology of designing intelligent systems***. Within such technology all the necessary partial technology should be harmonized, i.e. it **should be ensured compatibility of design solutions** which can be resulted within partial technologies.
  Compatibility of such design solutions means compatibility of different kinds of ***intelligent system components*** which, in general, can be the products of developing by different and independent developer teams. Among other factors it is necessary to ensure compatibility of different kinds of ***knowledge*** within knowledge base, different ***models for problem solving*** which are used by intellectual problem solver, compatibility of different ***models of understanding external information*** received by intelligent system on different channels, in different formats and on different languages.
- To create a general integrated technology of intelligent system design it is necessary to create a **general formal theory of intelligent system** [1].
- Intelligent system design technology should decrease work content not only while initial development but also while **permanent improvement** process (modernization, re-engineering) during the operation. In other words **intelligent systems** based on such technology **should be flexible**, easy modified, reconfigurable.
- Formal *models of intelligent systems* which are results of these systems design should be **simple as much as possible** and understandable not only by ***interpreters*** used for its implementation on different platforms but also by all the developers of such models.
- As different *intelligent systems* have, in general, different ***knowledge representation models*** and ***knowledge processing models*** the main problem of developing such models is creating a unified universal principle, a "skeleton", which makes it possible to build **hierarchical multilevel models of knowledge representation and processing with any configuration**.
  For *knowledge representation models* – this is move from *knowledge* to ***meta-knowledge***, from meta-knowledge to meta-meta-knowledge and so on, and, in particular, from description of actions (both internal and external) to description of actions on arbitrary higher level.
  For *knowledge processing models* – this is move from *agents* capable to execute one level actions to ***collective agents*** executing ***actions*** on arbitrary higher level.
  On the highest level internal activity of intelligent system represents balanced dialog of agents-optimists and agents-pessimists:
  - optimist generates ideas and hypothesis, looks for way out of various situations (contradictions) and for way of various problem solution;
  - pessimist always questions everything, always is searching for reasons and ideally – for proofs or retractions.

C. Analysis of Existing Approaches to Solving Specified Problems

Today there are a number of technologies for designing intelligent systems. Analysis of such technologies and corresponding tools is given in several works [25], [26]. Some of them have a significant impact on our research:

- Technology of designing real time expert systems on the base of instrumental system G2 [2];
- AT-Technology [25], [26];
- Technology of developing cloud intelligent services on the base of IACPaaS platform - Intelligent Application, Control and Platform as a Service [13];
- Technology of developing science portals [15];
- Technology of developing engineering knowledge portals which provide with complex engineering computations [9].

Today there are already a lot of modern technologies of designing intelligent systems. But they can solve not all problems mentioned above. Recently, most attention is focused on knowledge engineering and on technology of ontological engineering [8], [4] to the detriment of other equally important aspects of designing intelligent systems.

As a consequence, modern technologies of designing intelligent systems:

- are created not on the basis of **general formal theory of intelligent systems** and therefore do not consider detailed integration of "diverse" components of intelligent systems (***knowledge bases, knowledge processing machines, user interfaces***). Also modern technologies do not have a unified universal basis which allows within technology to integrate various scientific and practical results in area of artificial intelligence;
- do not provide compatibility of intelligent systems being developed and their components. This makes it difficult to organize simultaneous design of different components of the same system and the following integration of these components. Also it makes it difficult to develop ***collectives of intelligent systems***;
- do not provide **platform independence** of designing intelligent systems, i.e. clear separation of two processes: the process of development of complete *formal models of intelligent systems* and the process of development of these models interpreters on different platforms;
- do not have a formalized methodologies of complex **collective** designing intelligent systems, and in particular, do not have clearly formalized scope of full independence of simultaneously executed branches of designing and necessary points for coordination of these branches. Therefore do not provide saving both development time and labor;
- do not include ***methodology of training engineers of intelligent systems*** and, consequently, do not provide further training during development and the operation of these systems;
- do not support **their own development** including analysis and systematization of design experience.

D. The Proposed Approach to the Specified Problems

The following principles form the basis of the proposed approach to creating a *complex technology of designing intelligent systems*:

- Using ontological approach to intelligent system design, i.e. an approach based on hierarchical system of ***formal ontologies*** (on precisely specified systems of concepts which correspond to different and precisely distinguished levels of intelligent system consideration).
- Development of concepts and the corresponding ontologies for ***formal sense representation*** of any kind of knowledge, formal ontologies among them [21], [24], [29].
- Developing ***Formal ontology of intelligent systems*** which will be used as the base for unification and simplification for formal models of intelligent systems.
- Developing hierarchical system of consistent (compatible) ***formal ontologies*** for different kinds of *knowledge* and different *knowledge processing models*. It ensures integration of different kinds of knowledge and different models of knowledge processing, and also, independence from *platforms* of their interpretation. Developing ***General model of knowledge processing*** which represents a collective of agents working with common ***semantic memory***, interacting through this memory and controlled by knowledge kept in this memory. Formalizing this model as an ontology in relation to which different ontologies of specific knowledge processing models (inductive, deductive, crisp, fuzzy and so on) will be partial ontologies.
- Developing hierarchical system of consistent formal ontologies for design activity aiming not only to develop intelligent systems but also to permanently modify them during the operation. Such hierarchical system of ***formal ontologies of designing*** together with the corresponding tools should ensure a high level of improvement of intelligent systems during the operation.
- High level of flexibility of suggested technology is achieved due to the fact that this technology itself is implemented as an ***intelligent metasystem*** which ensures a complex support of intelligent system development in accordance with suggested technology and due to the fact that this technology itself is built on the same technology.
- Developing a ***formal ontology for improvement of technology of designing intelligent systems***: accumulation and systematization of design experience, replenishment of reusable component libraries and so on.
- Using ***methodology of component-based design*** on the base of permanently replenished ***library of reusable components***. This replenishment is carried out by developers of design technology as well as by developers of specific intelligent systems. Thus the proposed technology formalized as *intelligent metasystem* and implemented on the same technology has high rates of growth because it has:
  - effective tools for specification of design experience of engineers;
  - effective tools for specification of new scientific results (i.e. fundamentally new models, tools and methods for developing intelligent systems);

- effective tools for modifying actual models, tools and methods.

- Availability of unified foundation which allows to construct **different hierarchy levels of *intelligent systems components*** on its base, i.e. to move from level to meta-level, from knowledge – to meta-knowledge, from actions, action classes and ways to perform this – to meta-actions, meta-action classes and ways to perform this, from systems to metasystems.
  With this there is an opportunity to create multilevel libraries of reusable compatible components: libraries of knowledge and meta-knowledge, libraries of action and meta-action classes, libraries of ways to perform actions and meta-actions, libraries of typical base-level subsystems and subsystems of different meta-levels.
  All of this allow to **increase** substantially **the level of *component-based design*** – computer systems will be assembled not only from "small" but from "large-scale" components of any hierarchy level.
- Proposed technology represents an open semantic technology of component platform independent development of flexible compatible intelligent systems. We called this technology ***OSTIS Technology*** (Open Semantic Technology for Intelligent Systems). More detailed about base principles of this technology see in [11], [12].

E. Introduction to Ontology-based Design of Intelligent Systems

Since design is a key human activity ***General ontology of designing*** any kind of artefact is one of ***high level ontologies*** and represents a system of concepts which is the base of design activity systematization and control.

Design is a process of developing certain ***artefact*** information model (specification) sufficient to implement this artefact.

Ontology-based design is design on the base of using different ontologies and the result of such design is an ***ontological model of artefact being developed***, i.e. the model corresponding to ***ontology*** which describes characteristic of such artefacts.

As regard the ontology ***formalization*** in memory of ***intelligent system*** - ontologies should be considered the most important kind of ***knowledge*** used by intelligent systems. At the same time it is important to consider formalization not only of ***ontologies*** themselves (their internal structure) but also formalization of different kinds of connections and correspondences between ontologies. In particular it is very important to describe implicitly decomposition and semantic hierarchy of ontologies within ***knowledge base***. Also in this context it is important to describe connections of ontologies not only among themselves but also their connections with other kinds of ***knowledge***, for example, formal models of ***subject domains*** corresponding to ontologies, ontological models of different entities (in particular, designed artefacts).

This paper will consider ontology formalization on the base of ***semantic network***. It makes it possible to analyse more constructive the structure of ontologies themselves, the structure of ***subject domains*** and also connections and correspondences between them.

When considering ***General ontology of designing*** it makes sense to talk about a system of three interconnected ontologies as a minimum:

- ***General ontology of designing***
- ***General ontology of artefacts and their models***
- ***General ontology of intelligent systems for design automation***.

The first of these describes a general methodology of designing. The second – describes ontological models of designed objects. The third – describes information and instrumental designing tools.

Components of any technology including technology of designing intelligent systems are as follows:

- Ontology of corresponding class of artefacts being developed. This ontology describes how these artefacts and their ontological models are organized;
- Ontology of designing artefacts of specified class which describes methodology of designing artefacts of specified class;
- Ontological model of tools for designing artefacts of specified class.

It is clear that in addition to *General ontology of designing* there are a number of other partial ontologies of designing which make more precise (detail) general organizing principles of design activity taking into account the particularities of specific types of designed objects and the particularities of specific design stage organization.

For example, we can talk about ***Ontology of designing intelligent systems*** and about ***Ontology of testing intelligent systems***.

Each ontology is a specification of system of concepts used it the corresponding subject domain. Connection between ontologies and subject domains is specified by ***subject domain**** relation (to be a subject domain of specified ontology).

In terms of *Ontology of designing intelligent systems* which is the main subject of this paper we should talk about a system of ontologies and their corresponding subject domains which is presented on Fig. 1.

Subject domain of designing intelligent systems is a subject domain within which objects of research are processes of designing intelligent systems. It is important to construct this domain to minimize work content of designing processes and to provide a high quality of designed intelligent systems.

Such a designing quality ("cheap but good") could be achieved not only on the base of well thought-out unified *Ontology of designing intelligent systems* but also on the base of *Ontology of intelligent systems and their models* which clarifies and unifies the structure of designed intelligent systems.

For this purpose in proposed ***OSTIS Technology*** we have made clarification and **unification** of internal structure of designed intelligent systems.

Fig. 1 introduces a concept of ***ostis-system*** – intelligent system being developed on ***OSTIS Technology,*** and also a

concept of ***IMS Metasystem*** (Intelligent MetaSystem) which is the *ostis-system* for *ostis-systems* design automation.

Note also that information presented on Fig. 1 is a text on the language called ***SCg-code*** (Semantic Code Graphic) which is an universal language for visualizing ***ostis-systems knowledge bases***. The detailed description of *SCg-code* syntax and semantic see in [3].

II. Base Language for Sense Knowledge Representation

Quality of internal language for knowledge representation in memory of intelligent systems is the main factor which determines effectiveness of designed intelligent systems as well as effectiveness of design activity for developing these systems. Such language should be

- as simple as possible,
- universal,
- extensible, open.

The main problem of knowledge representation is the creation of such a language that would be convenient and easily interpretable by not only intelligent system but also by a person who can be a developer or a user of the system. That could be achieved only by one way – to get closer as much as possible to what is called ***sense knowledge representation***.

The main requirement to a ***formal language of sense knowledge representation*** is to eliminate semantic equivalence of texts within knowledge base of each intelligent system. So within knowledge base semantically equivalent texts should be transformed into the same text, i.e. such texts should not be duplicated.

So sense knowledge representation can be treated as invariant of variety of semantic forms of this knowledge representation. Note that semantically equivalent texts within knowledge base of different intelligent systems may exist but at the same time such texts should be completely structurally and semantically equivalent up to isomorphism.

Variety of syntactic forms (variants) of representation of the same information (the same knowledge) should be reduced to only one variant while creating a language of sense knowledge representation. Within this variant it would be sufficient simply to determine semantic equivalence of two texts using syntactic structure of these texts. Syntactic equivalence (isomorphism) of two texts should be the necessary condition of semantic equivalence of these texts.

The main guideline while creating a formal language of internal sense knowledge representation in intelligent system memory is to take away from language all unnecessary things unrelated to creating an internal model of external and internal "world" in which this intelligent system will solve various kinds of problems.

The essence of each language structure, its semantic power is specified (1) by a method of information encryption and (2) by a certain *ontology* which clarifies semantics of concepts used in this language. In its turn a method of information encryption is specified (1) by a method of representing signs of entities described by language texts and (2) by a method of describing connections between described entities (these connections in language texts are represented as syntactic connections between signs of specified entities).

Universal and open language of knowledge representation can be created only on the base of hierarchical system of ontologies within which the following are distinguished: a top-level base ontology and a family of ontologies which are partial in relation to this base ontology and which provide unlimited detailing for describing entities from this top-level base ontology.

As a base internal formal language for knowledge representation in memory of intelligent systems we suggest the language which is named ***SC-code*** (Semantic Computer Code).

Now consider **principles underlying SC-code** and also its features and advantages.

A. Specification of SC-code concept

Formally speaking ***SC-code*** is a set of texts (sc-texts) set-theoretical union of which represents an unlimited structure including descriptions of various entities. This unlimited structure is nothing but ***Subject domain of entities*** which contains descriptions of various entities on the initial level of their detailing. Specified subject domain is the highest level subject domain because there is no other subject domain in relation to which it would be ***partial subject domain****. As any other subject domain, *Subject domain of entities* has its own ontology corresponding to it, namely, ***Ontology of entities*** which specifies concepts used within *Subject domain of entities* and specifies SC-code semantic.

Fragments (substructures) of *Subject domain of entities* will be referred to as ***texts of SC-code*** or simply ***sc-texts***.

So language as a set of in some ways organized texts about certain subject domain is put in correspondence with integrated cognitive model of this subject domain.

Strictly speaking SC-code can be treated not as a language but as a universal code which provides a unified and therefore semantically compatible representation of various subject domains and their corresponding ontologies on the base of ***Subject domain and ontology of entities***.

Base nature of *Subject domains of entities* is reflected in the fact that all (!) other subject domains are not just ***partial**** in relation to it but are also its subsets, i.e. ***included**** structures. Classes of entities included in *Subject domain of entities* not just as key classes (key concepts) in different combinations within ***partial subject domains**** will serve as key concepts the sense of which is clarified within *ontologies* corresponding to these subject *domains*.

B. Concept of sc-element

All (!) syntactically elementary (atomic) fragments of SC-code texts (sc-texts) are ***signs*** of their corresponding (denoted by them) *entities*. Such elementary fragments of sc-texts we will name ***sc-elements***. It should be stressed, that:

- Elementary (atomic) character of sc-elements means that such elements do not have internal structure, i.e. do not consist of any other fragments of sc-texts as, for example,

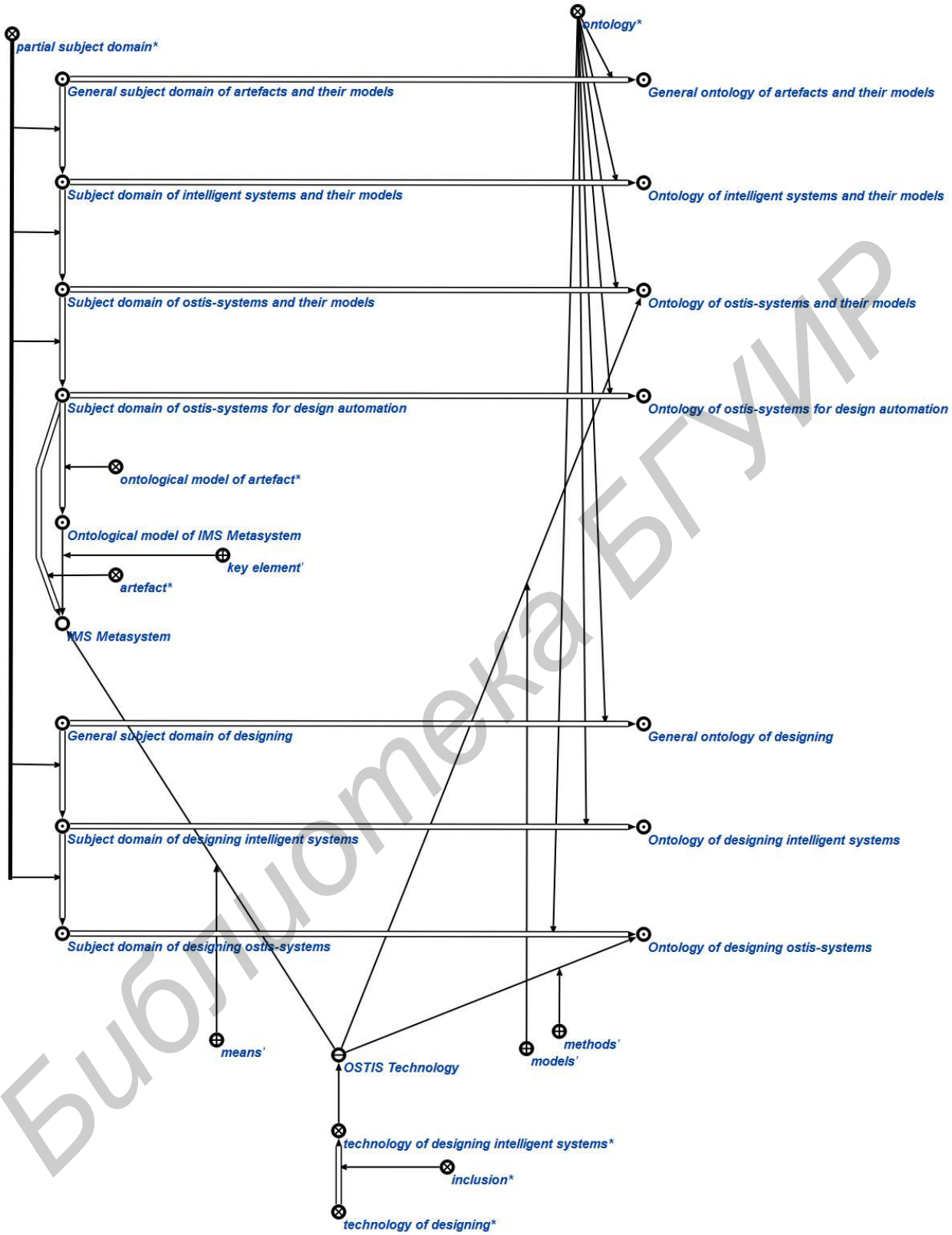

Figure 1. System of the ontologies and subject domains, connected with the Ontology of designing intelligent systems

signs of traditional languages which, in general, represent phrases and consist of words and, further, from letters;

- signs of any (!) entities can be presented by sc-elements. Thus SC-code is universal in the sense that sc-texts can describe any entities;
- sc-texts do not contain any signs beside sc-elements;

- within each sc-text, including sc-text of knowledge base of intelligent system, the following should not contained:
  - pairs of synonymic elements which denote the same entity. So sign of any described entity is contained only once in the corresponding sc-texts;
  - homonymous sc-elements, i.e. sc-elements each of which represents different entity while considering from different perspectives.
    Thus correspondence between sc-elements of given sc-text and entities which are described in this text is one-to-one correspondence.

C. Description of sc-element

Type of arbitrary considered sc-element, and consequently, type of entity denoted by this sc-element is specified within SC-code as follows:

- sc-element is introduced which denotes specified type of sc-elements, i.e. a class of those and only those sc-elements which correspond to this type;
- sc-element is introduced which denotes connection of membership of considered sc-element with that sc-element class which is denoted by sc-element introduced above.

D. Typology of sc-elements

Typology (classification) of sc-elements on the main criteria looks as follows.

On constance-variability criterion a set of sc-elements is split on:

- sc-constants (constant sc-elements);
- sc-variables (variable sc-elements).

On structure criterion (by a "place" of sc-element within sc-text structure) a set of constant sc-elements is split on:

- signs of external entities;
- signs of terminal abstract entities (i.e. abstract entities are not sets);
- signs of sets of sc-elements.

In its turn a set of signs of sets of sc-elements on structure criterion is split on:

- sc-classes – set of signs of classes of sc-elements;
- sc-links – set of signs of connections between sc-elements; each such connection is treated as a set of sc-elements connected by this connection;
- sc-structures – set of signs of structures consisting, in general, of sc-elements of various structure types.

Type of sc-variable is determined by the area of its possible values. It can be:

- variable sign of external entity if all of this variable values are signs of external entities;
- variable sc-link if all of this variable values are constant sc-links;
- and so on.

On ***temporal criterion*** set of sc-elements is split on:

- denotations of permanent entities;
- denotations of temporary entities.

E. SC-code syntax

SC-code syntax is defined as:

- collection of sc-element classes which are specified not with standard SC-code means described above but implicitly, syntactically, by means of assignment of corresponding "labels" to sc-elements. This is nothing but ***alphabet of sc-elements***;
- implicit (syntactic) representation of membership connections between signs of binary connections and components of these connections. This implicit representation of connections is specified by two incident relations of sc-elements: incident relation of signs of binary connections and sc-elements connected by these connections and incident relation of signs of binary oriented connections (sc-arcs) and its second components, i.e. those sc-elements for which these arcs are ingoing arcs.

Stress that all mentioned syntactic "techniques" of SC-code have a clear semantic interpretation – this is always implicit representation of membership connection, i.e. such a representation for which explicitly specifying this connection sc-element is not introduced. Such membership connection specifies (1) membership of sc-element to a given sc-elements class included in sc-element alphabet or (2) membership of sc-element to a given binary connection which is presented by sc-elements denoting this connection.

Thus SC-code does not have syntax in the traditional sense. SC-code syntax is:

- a special syntactic, implicit, form of distinguishing certain classes of sc-elements (syntactically distinguished classes which compose the language alphabet);
- a special syntactic, implicit, form of representation of membership connections as syntactically specified incident connections connecting sc-elements which denote binary connections with sc-elements which are components of these binary connections.

In purpose to formalize SC-code syntax within *Subject domain of entities* the following two concepts are introduced:

Syntactically specified class of elements
= such class of sc-elements membership in which for each given sc-element is specified not by membership pair but by means of addition of corresponding label to this sc-element. A set of label types and a family of syntactically specified classes of sc-elements are in one-to-one correspondence

sc-connector
= atomic binary link
= sc-element which denotes binary connection between sc-elements and for which its connection with components of this binary connection is specified syntactically by means of incidence relation
*<= partitioning**
- *sc-arc*
- *sc-edge*

The above formal text is a text of ***SCn-code*** (Semantic Code Natural) which provides with structured hypertext visu-

alization of *SC-code* texts. Detailed description of *SCn-code* syntax and semantics see in [3].

F. Concept of a set of sc-elements

Existing unity of syntax and semantic aspects of SC-code is reflected also in the fact that sc-elements which are signs of sets can denote only such sets elements of which are sc-elements, i.e. signs of various entities. Those sets will be called ***sc-sets***. Such sets are syntactic and semantic constructions simultaneously if, certainly, some semantic criteria will be taken into account while creating those sets.

If a set consists of signs of all those and only those entities which have a common characteristic this set will be a ***class*** of signs of equivalent entities.

If a set consists of signs of all those and only those entities which are interconnected by certain connection this set will be a ***link*** of signs of interconnected entities.

If a set consists of signs of all those and only those entities (it is possible also of signs of connections between these entities) representing a certain holistic construction which are independent object of research then this set will be a holistic ***structure*** of signs.

Stress that mentioned semantic restriction on set elements does not reduce semantic power of SC-code because any set of entities can be put in one-to-one correspondence with set of signs of these entities which in fact is information model of initial set.

It is important to note that within any significant ***sc-text*** the number of secondary sc-elements denoting sc-element sets greatly exceeds the number of primary (terminal) sc-elements which are not signs of sets (such sc-elements are signs of material entities, signs of such abstract entities as geometric point, as number).

Note also that any sc-text can be treated as hierarchical system of sets based on ***membership**** relation. This relation connects sc-elements identifying sets with sc-elements which are elements of these identified sets (in doing so the signs of certain sets can be elements of other sets). Therefore SC-code can be treated as a language with base set-theoretical semantic interpretation of its texts.

G. SC-code as semantic network language

Construction of sc-text can't be linear because sign of each described entity is included in sc-text (in knowledge base also) only once and because each described entity can be connected by unlimited number of connections with other entities. In its turn this means that each sign of sc-text can have unlimited number of connections with other signs. Such non-linear constructions are called ***semantic networks*** [5], [16].

Consequently SC-code is a language of semantic networks. The whole *Subject domain of entities* is, accordingly, an infinite semantic network which has integrated into itself all various texts of SC-code.

The main advantage of semantic networks and of SC-code texts, in particular, is consolidation of syntactic and semantic aspects of knowledge representation. It reduces significantly computational complexity of knowledge processing [23].

Stress that moving from traditional (linear) texts to semantic networks can be treated as a process of getting rid of that language excessiveness which are resulted from communicative function of traditional languages but are not necessary for creation of formal sense internal model of world in which intelligent system "lives".

Getting rid of specified excessiveness includes:

- exclusion of text fragments non-interpreted semantically – letters, separators, delimiters, words which are not signs of entities. All the atomic fragments of texts become signs;
- exclusion of synonymy of signs;
- exclusion of homonymy of signs.

H. SC-code as the Base of Representation of Various Subject Domains and Ontologies

SC-code is the base for creation of formal models of various *subject domains* and for representation of other kinds of knowledge. For these purposes it is introduced (1) sub-language of SC-code which is specified by ***Subject domain of subject domains*** and by the corresponding ***Ontology of subject domains***; (2) sub-language of SC-code which is specified by ***Subject domain of ontologies*** and by the corresponding ***Ontology of ontologies***; (3) a family of other sub-languages of SC-code oriented on representation of other kinds of knowledge.

Thus SC-code results all the variety of knowledge kinds not only to a common syntactic form but also to one common high level ontology, ***Ontology of entities***, which is the basis for syntactic structure of SC-code texts as well as base semantic interpretation of these texts.

I. SC-code and the System of Specialized Languages

Formal language of internal sense knowledge representation in memory of intelligent systems represents an integrated, open and permanently developed language based on SC-code. In its entirely this language represents a hierarchical system of specialized formal languages each of which is a sub-language of SC-code and is specified (1) by subject domain ***partial**** in relation to *Subject domain of entities* and (2) by ontology which specifies a set of concepts used by this partial subject domain and which itself is ***partial**** in relation to *Ontology of entities*.

Since integrated language of internal sense knowledge representation is a hierarchical system of languages created on SC-code it is based on:

- language of formal representation of subject domains which is specified by *Subject domain and Ontology of subject domains*;
- language of formal representation of ontologies which is specified by *Subject domain and Ontology of ontologies*;
- base language SC-code itself which is specified by *Subject domain and Ontology of entities*.

III. Formal Models of Subject Domains

In this chapter the following topics will be considered:

- general principles for representation on ***SC-code*** of formal models of ***subject domains***;
- formal model of ***Subject domain of entities*** which corresponds to ***SC-code***;
- formal model of ***Subject domain of subject domains*** for which researched objects are various subject domains.

Explicit distinguishing various *subject domains* within knowledge bases of intelligent systems, and in particular, *subject domains of actions and tasks*, makes it possible to localize search for ways to solve specific problems. These problems could be solved by intelligent systems themselves or by system users with help of intelligent system.

A. Structure of Formal Models of Subject Domains

Formal model ***Subject domain*** represented on SC-code is sc-structure within which by means of special collection of ***role relations*** several key *elements* of this structure are distinguished and roles of these key elements within this structure are specified. Such role relations are subsets of *Membership relation*.

First of all key elements of subject domain are signs of *considered concepts* clarification of sense of which are essential for semantic analysis of specified subject domain. Also key elements of subject domain can be signs of those objects of research within this subject domain which have special characteristics and used for description of sense of above key concepts within this subject domain. Number 0 and number 1 are examples of such *key researched objects* within *Subject domain of numbers*.

Description of sense of such key concepts of subject domain, i.e. specification, is nothing but ***ontology*** corresponding to specified *subject domain*.

Roles of concepts contained in subject domain follow:

considered concept'
= *concept considered in given subject domain'*
= *key concept of given subject domain'*
<= *partitioning**
- *the researched concept'*
- *non-researched but considered concept'*

<= *partitioning**
- *concept introduced in given subject domain'*
- *considered concept introduced in other subject domain'*

There are four variants of clarifying roles for concepts considered in subject domains:

- concept can be researched and introduced in given subject domain;
- concept can be researched in given subject domain but introduced in other subject domain;
- concept can be non-researched but introduced in given subject domain;
- concept can be non-researched and non-introduced in given subject domain.

The above-mentioned ***role relations*** specify different correspondences between *subject domains* and ***concepts*** considered in these domains, for example, the correspondence between concepts and subject domains where these concepts are researched.

It is not hard to see that these role relations describe semantic distribution of concepts between subject domains. In general, this distribution takes into account differences in concepts usage (considering) in different subject domains, and also it takes into account that one and the same concept can be used in different subject domains.

It is clear that subject domains considering one and the same concept have a deep semantic relationship between themselves.

More detailed consideration of role relation ***the researched concept'*** taking into account a structure type of these concepts allows to specify the following sub-classes of these relations:

researched concept'
= *to be the researched concept'*
⊃ *class of primary researched objects'*
  ⊃ *the maximum class of primary researched objects'*
    = *class of primary researched objects for which there is no other class of primary researched objects that would be its subset within subject domain'*
⊃ *the researched relation'*
  = *class of researched links'*
  ⊃ *maximum researched relation'*
⊃ *class of researched structures'*
  ⊃ *maximum class of researched structures'*
⊃ *class of researched classes'*
  = *parameter (characteristic) specified on a set of researched objects'*

Set-theoretical connection between a concept considered in a subject domain and the set of elements of this subject domain is specified by the following role relations:

- *a set all of elements of which are located in given subject domain'*
- *a set not all of elements of which are located in given subject domain'*

Semantic hierarchy of subject domains is specified by the following role relations:

- *a concept which is an instance of a concept researched in other subject domain'*
- *a concept which is a subset of a concept researched in other subject domain'*
- *a concept for which in other subject domain a researched concept exists against which the first specified concept is a class of parts of its instances'*

B. Concept System of Subject Domain of Entities

Let's consider ***Subject domain of entities*** which on the base level specifies syntax and semantic of ***SC-code*** and which is *subject domain* of the highest possible level. It follows that concepts considered in this subject domain can't be introduced in other *subject domains* because to make that possible these

other subject domains would be higher level subject domains. The highest possible level of *Subject domain of entities* means also that a lot of concepts will be only introduced in this subject domain and will be researched in other *subject domains* as ***classes of primary researched objects'*** including. Within *Subject domain of entities* ***maximum class of primary researched objects'*** is a concept of ***entity***, more precise, a concept of set of signs of various entities which is identical with ***sc-element*** within ***SC-code***.

Classification of a set of ***sc-elements*** on different criteria produces a hierarchical system of subclasses of maximum class of sc-elements. A feature of *Subject domain of entities* is the fact that signs of all the specified partial classes of sc-elements are themselves *sc-elements*, i.e. instances of ***maximum class of primary researched objects'***. Moreover within considered subject domain all the ***researched relations'***, all the ***classes of researched structures'*** and all the ***classes of researched classes'*** are also partial ***classes of primary researched objects'***.

Let's consider three general criteria of sc-elements classification:

- logical typology *of sc-elements*;
- structural typology *of sc-elements*;
- temporal typology *of sc-elements*.

sc-element
= abstract sign of certain entity for which its internal structure is not important but only its conditional connection with entity denoted by this sign is important
<= partitioning:*
- *sc-constant*
- *sc-variable*

<= partitioning:*
- *terminal sc-element*
- *sc-set*
  <= partitioning:*
  - *sc-link*
  - *sc-structure*
  - *sc-class*

<= partitioning:*
- *denotation of permanent entity*
- *denotation of temporary entity*

Let's consider more detailed classes of researched objects introduced within *Subject domain of entities*.

sc-constant
<= partitioning:*
- *terminal sc-constant*
  = sc-constant which is not a sign of set
- *constant sc-set*
  = sc-element denoting specific set of sc-elements
  <= partitioning:*
  - *constant sc-link*
  - *constant sc-structure*
  - *constant sc-class*
    <= partitioning:*
    - *constant class of terminal sc-constants*
    - *constant sc-relation*
      = constant class of constant sc-links
    - *constant class of constant sc-structures*
    - *constant class of constant sc-classes*
    - *constant class of sc-variables*

<= partitioning:*
- *constant permanent entity*
  = permanently existing constant entity
- *constant temporary entity*
  = temporarily existing constant entity

constant sc-link
<= partitioning:*
- *constant binary sc-link*
- *constant non-binary sc-link*

<= partitioning:*
- *constant non-oriented sc-link*
- *constant oriented sc-link*

<= partitioning:*
- *constant sc-multilink*
  = constant sc-link with multiple occurrence of some components
- *constant sc-link with single occurrence of all its components*

<= partitioning:*
- *constant sc-metalink*
  = constant sc-link which has some other sc-links as its components
- *constant sc-link which does not have other sc-links as its components*

constant binary sc-link
<= partitioning:*
- *constant sc-link about membership nature*
- *constant binary sc-link which is not about membership nature*

constant sc-link about membership nature
<= partitioning:*
- *constant sc-link of membership*
  = Membership relation
- *constant sc-link of non-membership*
  = Non-membership relation
- *constant sc-link of fuzzy membership*

sc-variable
= sc-element denoting arbitrary sc-element from certain set of sc-elements which are possible values of this arbitrary sc-element

Thus sc-variables as well as sc-sets can be treated as secondary sc-elements (secondary signs) because each sc-set is a sign of set of sc-element and each sc-variable "runs" certain set of sc-elements representing a set of possible values of this variable.

sc-variable
<= partitioning:*
- *sc-variable with values which are sc-elements of one logical level*
  <= partitioning:*
  - *sc-variable of the 1st level*
    = sc-variable with values which are only sc-constants

- *sc-variable of the 2nd level*
  = sc-variable with values which are only sc-variables of the 1st level
- *sc-variables with values which are sc-elements of different logical levels*
  <= partitioning:*
  - *terminal sc-variable*
  - *variable sc-set*
    = sc-variables with values which are only sc-sets
    <= partitioning:*
    - *variable sc-link*
    - *variable sc-structure*
    - *variable sc-class*

  <= partitioning:*
  - *variable permanent entity*
  - *variable temporary entity*

terminal sc-element
<= partitioning:*
- *terminal sc-constant*
- *terminal sc-variable*
  <= partitioning:*
  - *terminal sc-variable of the 1st level*
    = sc-variable with values which are only terminal sc-constants
  - *terminal sc-variable of the 2nd level*
    = sc-variable with values which are only terminal sc-variables of the 1st level

sc-set
<= partitioning:*
- *constant sc-set*
- *variable sc-set*
  <= partitioning:*
  - *variable sc-set of the 1st level*
    = sc-variable with values which are only signs of constant sc-sets
  - *variable sc-set of the 2nd level*

sc-link
<= partitioning:*
- *constant sc-link*
- *variable sc-link*
  <= partitioning:*
  - *variable sc-link of the 1st level*
  - *variable sc-link of the 2nd level*

<= partitioning:*
- *binary sc-link*
  <= partitioning:*
  - *binary sc-link about membership nature*
    <= partitioning:*
    - *sc-link of membership*
    - *sc-link of non-membership*
    - *sc-link of fuzzy membership*
  - *binary sc-link which is not about membership nature*

sc-structure
<= partitioning:*
- *constant sc-structure*
- *variable sc-structure*
  <= partitioning:*
  - *variable sc-structure of the 1st level*
  - *variable sc-structure of the 2nd level*

sc-class
<= partitioning:*
- *constant sc-class*
- *variable sc-class*
  <= partitioning:*
  - *variable sc-class of the 1st level*
  - *variable sc-class of the 2nd level*

denotation of permanent entity
<= partitioning:*
- *constant permanent entity*
- *variable permanent entity*
  <= partitioning:*
  - *variable permanent entity of the 1st level*
  - *variable permanent entity of the 2nd level*

denotation of temporary entity
<= partitioning:*
- *constant temporary entity*
- *variable temporary entity*
  <= partitioning:*
  - *variable temporary entity of the 1st level*
  - *variable temporary entity of the 2nd level*

1) Consider SC-code means used for description of syntax structure of SC-code texts detailing:

syntactic text structure*
*= Relation connecting a sign of certain text (not necessarily sc-text) with sc-text describing its syntactic structure**

Alphabet of sc-elements
= Family of syntactically distinguished classes of sc-elements
= syntactically distinguished class of sc-elements
∋ sc-constant
= constant sc-element
∋ sc-variable
= variable sc-element
∋ sc-connector
= atomic sc-link
= sc-link connected implicitly (syntactically) with its components by means of ***Incidence relation of sc-elements*** *and* ***Incidence relation of sc-arcs with its second components*** *(i.e. with sc-elements these sc-arcs enter in)*
= syntactically distinguished class of binary links for which connection with their components is formalized by means of syntactically implemented incidence relations
⊂ binary sc-link
∋ sc-node
= sc-element which is not sc-connector
∋ sc-arc
∋ denotation of permanent entity
∋ denotation of temporary entity
∋ sc-link of membership
∋ sc-link of non-membership
∋ sc-link of fuzzy membership

Stress that each ***sc-element*** should have three labels as minimum which specify: (1) its logical type – constant or variable, (2) its structural type – sc-node or sc-connector, (3) its temporal type – permanent or temporary nature. Moreover sc-links about membership nature should have one more label specifying membership, non-membership, fuzzy membership.

Family of syntactically formalized relations on sc-elements
= *syntactically formalized relations on sc-elements*
∋ *label'*
  = *Relation which connects syntactically distinguished classes of sc-elements with instances of these classes'*
∋ *incidence of sc-connectors with their components'*
  ⊃ *incidence of sc-arcs with their second components'*
∋ *incidence of sc-arcs with their second components'*

Here is an example of how syntactic structure of *sc-texts* can be formalized with *SC-code* means and used to convert *sc-texts* to more compact unified form based on appeared in knowledge base information clarifying the sense of certain *sc-elements*. Fig. 2 demonstrates certain constant *sc-node* ***ci***. Let it became known after some time that this sc-node is a sign of *binary sc-link** and components of this link also became known (see Fig. 3). Then semantic structure of this *sc-text* can be converted to the form presented on Fig. 4 and further this *sc-text* can be represented in more compact form demonstrated on Fig. 5. SC-code text on Fig. 4 is a description of syntactic structure of *sc-text* from Fig. 5 and therefore these texts can be connected by considered above relation to be ***syntactic text structure****.

It's clear that *sc-texts* describing syntactic structure of other *sc-texts* should not be kept in knowledge base. They can appear in knowledge base only for the period of analysis and updating syntactic structure (coding way) of some knowledge base fragments to make more compact and unified the way of coding these fragments.

2) Subject domain of entities comprises a number of relations which has inter-subject nature.: In the corresponding partial subject domains various clarification of such relations (i.e. various relations which are their subsets) are introduced and researched. Examples of such inter-subject relations follows:

part*
= *to be a part**

generalized part*
= *connection of class of entities with class of certain kind of parts of these entities**

decomposition*
= *splitting a specified entity into a set of its parts (components)**
= *connection of entity with maximum (complete) family of their disjoint parts**

generalized decomposition*
= *connection of class of entities with maximum family of classes of disjoint parts of these entities**

We give examples of clarifying inter-subject relations for different subject domains.

Within *Subject domain of sets* the following clarifications of specified relations are introduced:

inclusion*
= *subset**
⊂ *part**

partitioning*
= *decomposition of a set into maximum family its pairwise disjoint subsets**
⊂ *decomposition**

Within *Subject domain of geometric points and figures* the following clarification of *decomposition** relation is introduced:

decomposition of geometric figure*
= *set-theoretical union of geometric figures each pair of which either does not intersect or intersects but only by its boundary points**
⊂ *decomposition**

Within *Subject domain of temporary entities* the following clarifications of specified relations are introduced:

temporal part*
= *period of existence of specified temporary entity**
⊂ *part**

temporal decomposition*
= *decomposition of temporary entity into family of its disjoint temporal parts**
⊂ *decomposition**

C. Concept System of Subject Domain of Subject Domains

Consider formal model of one more important subject domain – ***Subject domain of subject domains*** within which various subject domains including this domain itself are objects of research.

Key concepts within ***Subject domain of subject domains*** are the following:

- ***role relations*** connecting signs of subject domains with sc-elements which are included in these *subject domains* and, in particular, with signs of key concepts of specified subject domains. Such *role relations* have been considered above;
- concepts which denote various ***classes of subject domain***;
- relations specified on *set of subject domain*;
- family of concepts which can be defined only within *Subject domain of subject domains*.

1) Consider certain classes of subject domains: which are important for intelligent systems.

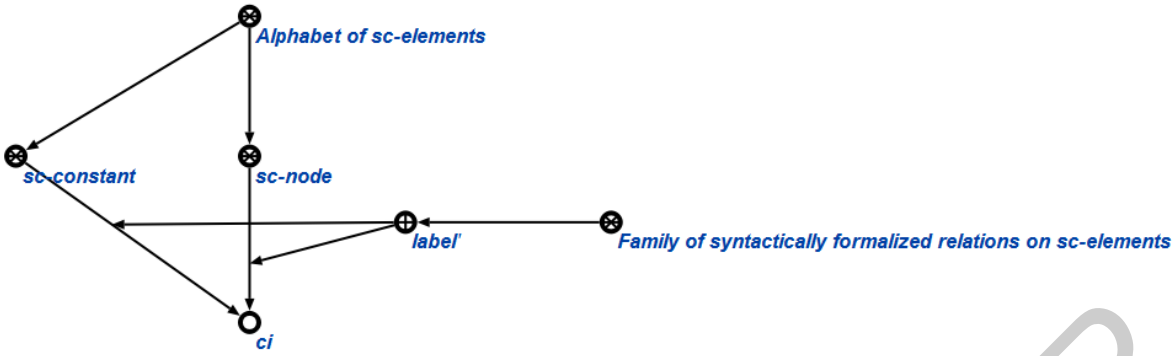

Figure 2. Specification of some constant sc-node ***ci***

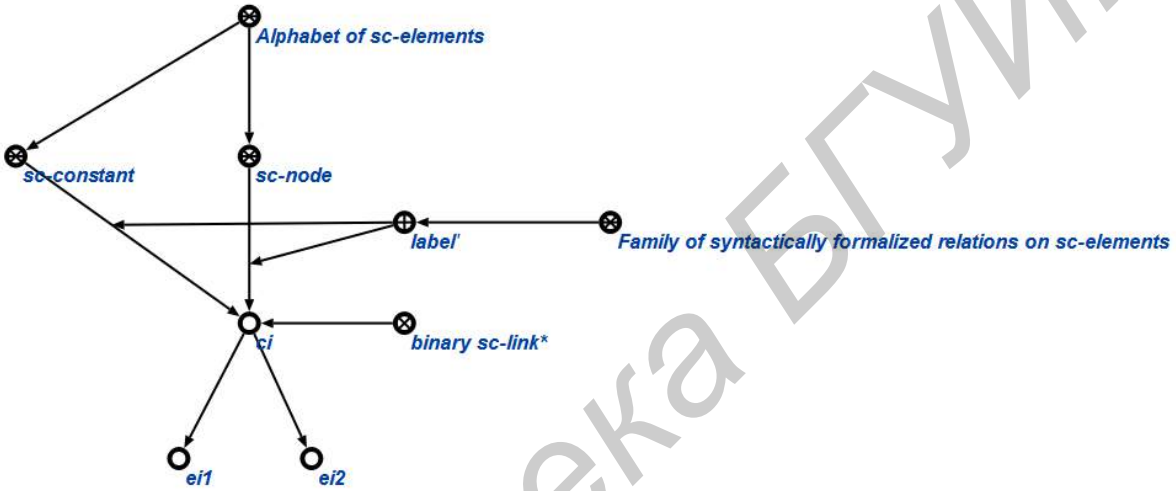

Figure 3. Adjustment of specification of sc-node ***ci***

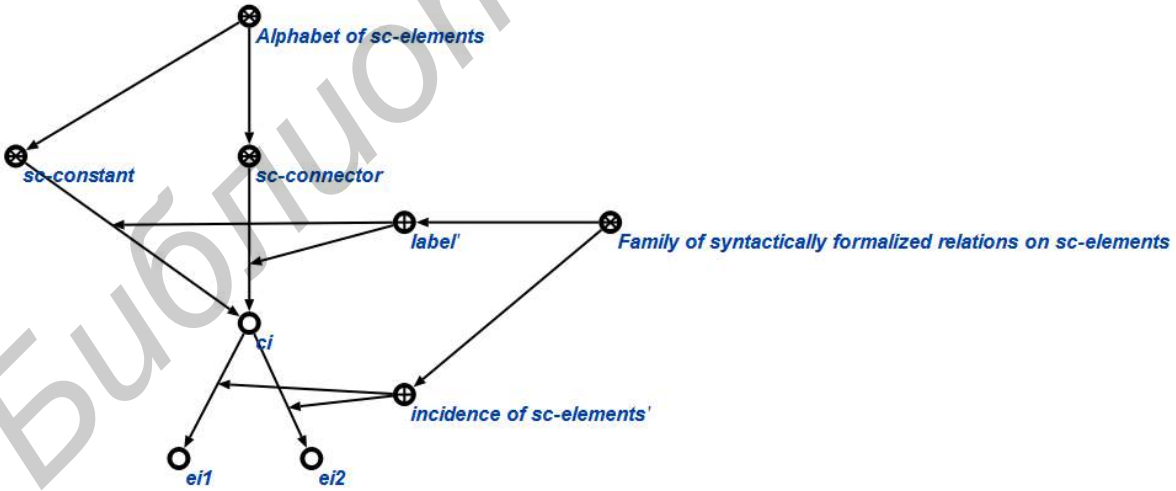

Figure 4. Change of syntax type of sc-node ***ci***

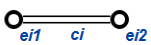

Figure 5. Transformation of sc-node ***ci*** into an sc-connector

subject domain
⊂ *sc-structure*
<= *partitioning*:*
- *static subject domain*
  ⊂ *static sc-structure*
- *dynamic subject domain*
  ⊂ *sc-process*
  = *dynamic sc-structure*
  ⊃ *subject domain of actions and tasks*

subject domain of actions and tasks
⊃ *subject domain of knowledge elicitation from specified subject domain and its ontology*
= *subject domain of actions for knowledge elicitation from current state of kept in memory fragment of specified subject domain and its ontology*
= *subject domain of activity during operation of specified subject domain and its ontology, i.e. generating answers to information questions within this subject domain and its ontology*
⊃ *subject domain of improvement of specified subject domain and its ontology*
= *subject domain of design actions for improvement of current state of kept in memory fragment of specified subject domain and its ontology*
⊃ *subject domain of designing on the base of specified subject domain and ontology of corresponding class of artefacts*
= *subject domain of design actions for creating information model (specification) of certain new artefact on the base of specified subject domain and ontology of corresponding class of artefacts*
⊃ *subject domain of external behaviour*
= *subject domain for changing state of specified dynamic subject domain*
⊃ *subject domain for implementation of designed artefacts*

The main problem which is solved within *subject domain of external behaviour* and its ontology – is the problem of behaviour planning, i.e. the problem of creating plan of action to achieve the specified goal.

2) Consider certain ***relations specified on set of subject domains****:*

relation specified on set of subject domains
= *relation connecting subject domains either among themselves or with other kinds of entities*
∋ *subject domain of knowledge elicitation from subject domain and its ontology**
= *Relation each link of which connects a subject domain with other subject domain describing actions and tasks for knowledge elicitation from kept in memory current state of the first subject domain and its ontology**
∋ *subject domain of improvement of subject domain and its ontology**
= *Relation each link of which connects a subject domain with other subject domain describing actions and tasks for improvement of the first subject domain and its ontology**
∋ *subject domain of designing**
= *Relation each link of which connects a dynamic subject domain describing a class of artefacts with other subject domain describing actions and tasks for developing information model of a new artefact belonging to specified class of artefacts**
∋ *subject domain of external behaviour**
= *Relation each link of which connects a dynamic subject domain with other subject domain describing actions and tasks for changing state of the first subject domain**
∋ *subject domain of implementation of designed artefacts**
⊂ *subject domain of external behaviour**
= *Relation each link of which connects a dynamic subject domain describing a class of artefacts with other subject domain describing actions and tasks for implementation (reproduction) of a new artefact belonging to specified class of artefacts**
∋ *partial subject domain**
= *to be a subject domain a set of researched objects of which is included in a set of researched objects of other specified subject domain**
= *subject domain of a subset of researched objects of specified subject domain**
∋ *subject domain with united set of researched objects**
= *to be a subject domain a set of researched objects of which is a union of sets of researched objects of certain family of other specified subject domains**
∋ *subject domain of class of parts of researched objects of specified subject domain**
= *to be a subject domain a set of researched objects of which is a class of parts of researched objects of specified subject domain* (specified parts can be either spatial or temporal)*
∋ *subject domains equivalent over a set of researched objects**
= *subject domains which have the same researched objects but do not consider different connections of these researched objects between themselves as well as with other entities**
∋ *ontology**
= *to be an ontology for the specified subject domain**
= *Relation each link of which connects sign of a subject domain with sign of the corresponding ontology**

3) At the end of consideration of Subject domain of subject domains: let's give an example of concepts which can be defined only within this subject domain. In particular, such a concept is a concept of ***concept***.

concept
= *class of entities which at least within one of subject domains implements the role of researched class (the role of class of researched objects, the role of researched relation, the role of researched class of structures, the role of researched class of classes)*
⊂ *sc-class*

So far from each class of entities can have a status of ***concept***.

IV. FORMAL MODELS OF ONTOLOGIES

This chapter addresses some aspects of formal ***ontologies*** representation using the tools of basic language of sense knowledge representation, i.e. formal ontologies representation as *sc-texts*.

Semantics of any language each of which is a set of sign constructions is specified as follows:

- on the lower level – by clarifying ***sign*** concept and by clarifying relations specified on sign language (this is SC-code level);
- on the top level – by hierarchical system of *subject domains* and the corresponding ***ontologies***. This system clarifies a variety of used in this language ***concepts*** distributing these concepts between ***subject domains*** and specifying them within the corresponding ontologies.

Explicit distinguishing ***ontologies*** in knowledge bases of intelligent systems is necessary for the following:

- to fix a coordinated current version of treatment (clarification) of all the used concepts;
- to secure a clear organization of continuous process of developing and coordinating the system of used concepts. In its turn, this requires detailed documenting (logging) all the changes in concept system.

Stress that while designing knowledge bases it is necessary on every stage to secure *semantic compatibility* of knowledge bases and their components. This is especially important when different collectives of knowledge engineers participate in the same development process and when the system of used concepts is being changed constantly and therefore ontologies also is being changed constantly. It is clear, that such compatibility could not be achieved without explicit distinguishing ontologies, without *logs of coordinated changes* of different subject domain ontologies, without explicit distinguishing *coordinated specified versions* for each of ontologies.

Each ***ontology*** is a model (specification) of a ***subject domain***, more precisely, a specification of system of concepts used within this subject domain.

Classification criteria of ontologies are the following:

- class of subject domain specified be this ontology;
- type of specification itself of subject domain.

To type of specification of subject domain could be attributed the following:

- structural specifications of subject domains – ***structural ontologies***;
- set-theoretical specifications of subject domains – ***set-theoretical ontologies***;
- logical specifications of subject domain – ***logical ontologies***;
- terminological specifications of subject domains – ***terminological ontologies***;
- ***integrated ontologies*** which unite all the specified partial types of ontologies.

At the same time if we unite all ontologies of certain type in a integrated text then this text can be treated as a subject domain comprising various ontologies of specified type.

Each ***structural ontology*** represents a fragment of ***Subject domain of subject domains***. This fragment contains all information about specified subject domain:

- role structure of subject domain,
- typology of subject domain,
- connections of subject domain with other subject domains.

Each ***set-theoretical ontology*** comprises not only specification of concepts considered in the specified subject domain but also specification of connections between relations and their definition areas, between relations and their domains, between characteristics (parameters) and a set of those entities which possess these characteristics. This ontology comprises also description of other set-theoretical connections between concepts included in specified subject domain.

All ***set-theoretical ontologies*** can be put in correspondence with ***Subject domain of sets*** researched objects of which are various sets including concepts comprised by specified subject domains.

Each ***logical ontology*** comprises:

- indication of not-defined concepts of specified subject domain;
- definitions of defined concepts;
- description of hierarchical system of concepts based on the facts which concepts are used in definition of each concept;
- axioms;
- theorems;
- proofs;
- description of hierarchical system of axioms and theorems indicating which axioms and theorems have been used for proving each theorem;
- description of different types of connections and analogies between definitions, axioms, theorems and proofs.

By analogy with relationship between *structural ontology* and *Subject domain of subject domains* ***logical ontilogies*** is put in correspondence with ***Subject domain of logical formulas*** key concepts of which are the following: ***sc-variable*** concept, concepts ***of logical formula, atomic logical formula, non-atomic formula of existence, conjunctive formula, disjunctive formula, implicative formula, formula of negation*** and other concepts.

Each ***atomic logical formula*** is treated on SC-code as a permanent *sc-scructure* among the elements of which there are both *sc-constants* and *sc-variables*. Each ***non-atomic logical formula*** is treated as *sc-link* which is attributed to the corresponding *sc-class of non-atomic logical formulas* and elements of which are signs of logical formulas which are components of given *non-atomic logical formula*.

Each t***erminological ontology*** comprises:

- description of main terms used for external representation of all the concepts considered by specified subject domain;
- description of minor terms belonging to different languages with specifying synonymy and homonymy;
- description of origin of used terms;

- description of connections of terms with authoritative documents in which these terms are used;
- description of linguistic specification of each term;
- description of rules of creating names for instances of concepts considered within a specified subject domain.

Terminological ontologies are put in correspondence with ***Subject domain of terms***, by analogy with relationship between *logical ontologies* and *Subject domain of logical formulas*.

It is clear that each considered in this chapter *subject domain* is put in correspondence with its *integrated ontology*:

- ***Ontology of entities***,
- ***Ontology of subject domains***,
- ***Ontology of sets***,
- ***Ontology of logical formulas***,
- ***Ontology of terms***.

Ontology of entities
= *Base ontology of SC-code reflecting main principles of its syntax and semantics*
= *Ontology of Subject domain of entities*
= *General ontology of entities*

V. Ontological Model of Intelligent System

A feature of designing knowledge data bases is the fact that knowledge bases are both objects and results of design because in this case information model of designed object which is the result of design and design object itself coincide, i.e. knowledge base and its information model are one and the same thing.

Knowledge bases of intelligent systems which are developed on the base of SC-code in the form of sc-models of knowledge bases and designing of those knowledge bases have a number of advantages and features.

Since SC-code is universal language of formal knowledge representation it can be used not only for representation of ontological model of knowledge base of intelligent system (sc-model of knowledge base) but also of ontological model (sc-model) of intelligent system on the whole. Clear that *sc-model of knowledge base* of intelligent system will be a part of sc-model of this intelligent system.

If we introduce the concept ***of extended knowledge base of intelligent system*** and if we consider sc-model of this intelligent system as such knowledge base we will have a lot of advantages. These advantages mainly relate to significant increasing flexibility of developed intelligent systems and of their platform independence. Flexibility level of developed intelligent systems is determined by work content for changing system maintaining its integrity. In case of collective development flexibility level is determined also work content is.

Platform independence level is determined by a number of ”coordination points” between developers of sc-models of certain intelligent systems (i.e. designers of such systems) and developers of interpreters for different platforms providing interpretation of sc-models of any intelligent systems. Ontological character of sc-models of intelligent systems provides an opportunity to clear indicate ”coordination points” which are concepts of the corresponding ontologies.

The structure ***of sc-model of extended knowledge base of ostis-system*** is a reflection of this ***ostis-system*** architecture because such extended knowledge base comprises:

- sc-model of ***main subsystem of ostis-system***;
- sc-model of ***subsystem of improvement of extended knowledge base*** of considered ostis-system***;***
- sc-model of ***subsystem of improvement of integrated knowledge processing machine*** of considered ostis-system. This integrated machine comprises knowledge processing machines of all subsystems of this ostis-system;
- sc-model of ***subsystem of improvement of integrated user interface*** of considered ostis-system. Such interface comprises user interfaces if all subsystems of this ostis-system.

All the specified subsystems of considered ostis-system, in their turn, comprise the following:

- ***sc-model of knowledge base*** of subsystem;
- ***sc-model of knowledge processing machine*** of subsystem;
- ***sc-model of user interface*** of subsystem.

Stress that *sc-model of user interface* of each subsystem of ostis-system and also *sc-model of integrated user interface* of the whole ostis-system consists of ***sc-model of knowledge base of user interface*** and ***sc-model of knowledge processing machine of user interface***.

sc-model of knowledge base of user interface, in its turn, comprises:

- description of syntax and semantics of all used external languages. This description should be full enough to make it possible translation of *SC-code* texts to external language and vise versa by *knowledge processing machine of user interface*;
- description of how “main window” of user interface of the subsystem of considered ostis-system is included in “main window” of *integrated user interface* of this ostis-system;
- description of interaction principles of user interface with users on the low interface level;
- interface models of users containing information about their peculiarities, possibilities and preferences. This information makes interface to be able to adapt to each user and to make more effective interaction with user.

So designing *ostis-system*, in fact, is reduced to designing its *extended knowledge base*. At the same time specific of designing *integrated knowledge processing machine of ostis-system* and its *integrated user interface* is reduced to only semantic specifics of the corresponding sc-models – of *sc-models of knowledge processing machines* and *sc-models of user interfaces* which are specified by the corresponding ontologies: ***Ontology of knowledge processing machines*** and ***Ontology of user interfaces***.

It follows that *Subject domain and ontology of sc-models of knowledge bases* coincide completely within OSTIS Tech-

nology with *Subject domain and ontology of sc-models of ostis-systems*.

Accordingly *Subject domain and ontology of designing sc-models of knowledge bases* is identified with *Subject domain and ontology of designing sc-models of ostis-systems*.

At the same time *Subject domain and ontology of sc-models of knowledge processing machines* and also *Subject domain and ontology of sc-models of user interfaces* become *partial** in relation to *Subject domain and ontology of sc-models of knowledge bases*.

Information about different kinds of activity of *ostis-system*, its subsystems and its users is also included in *sc-model of extended knowledge base of ostis-system*:

- information about work of *ostis-system users* on the low (front-end) level. This is results of monitoring and analysis of user activity on front-end level;
- information about *operation of ostis-system* includes knowledge elicitation (first of all by users) from the current state of *extended knowledge base*. Such knowledge elicitation means not only information search but also solution of problems of any level of complexity. Description of operational activity is formalized as logs of *dialogue between ostis-system and their users*. These logs (1) can be interesting for users themselves, (2) can be useful for clarifying models of information necessities of users – it is important for increasing efficiency of interconnecting with users on meaningful level, (3) can be useful for detecting errors and defects of ostis-system itself;
- information about continuous activity for *improving knowledge base of ostis-system* both in whole and within improving sc-models of its separate subsystems.

Information about improvement of knowledge base comprises:

- current coordinated state of knowledge base;
- log of changes of knowledge base;
- suggestions for improvement of knowledge base;
- current state of process of coordinating suggestions for improvement knowledge base;
- log of coordinating suggestions for improvement knowledge base.

So extended knowledge base of ostis-system reflects two viewpoints of ostis-system consideration:

- *ostis-system* architecture;
- *ostis-system* dynamics in terms of its operation and evolution (past activities and their results, current activities, future planned events).

VI. Ontological model of designing intelligent system

As it was mentioned in the previous chapter designing intelligent system using *OSTIS Technology* is reduced to design of its *extended knowledge base* which represents meaningful *sc-model* of developed intelligent system. Typology of *design actions* while designing intelligent system is determined by:

- typology of designed component of intelligent system;
- type of design action.

Depending on typology of designed component the following actions can be distinguished:

- subject-independent design actions for developing sc-models of various knowledge base fragments;
- specific actions for developing *sc-models of knowledge processing machines*;
- specific actions for developing *sc-models of user interfaces*.

Depending on type of design actions the following activities can be distinguished:

- testing of given fragment of extended knowledge base;
- elimination of contradictions and errors in knowledge base detected during testing;
- information waste removal (i.e. those sc-texts which are not needed more);
- addition of new *sc-texts* which do not change structure of any *subject domain* (i.e. its *ontology*);
- change of structure of *subject domain*. Each such change should have a clear specified transition period at the end of which all the needed corrections of this *subject domain* and its *ontology* caused by replacement of one group of concepts to another should be completed;
- whole complex of design actions for changing *subject domain* and its *ontology* in accordance with coordinated changes in system of concepts of this *subject domain*. The first step of this process is generating design action plan which is directed to knowledge base engineers. On the next step engineers of knowledge bases enter into knowledge base formal *definitions* of all new concepts and also of obsolete replaceable concepts on the base of currently used concepts. Further design actions can be completed manually or automatically;
- expertise, coordination and approval of suggestions for improvement of knowledge base.

VII. Intelligent System for Complex Automation of Designing Intelligent systems

Proposed *OSTIS Technology* is implemented as an intelligent system which created with OSTIS Technology itself. We called this system ***IMS Metasystem*** (Intelligent MetaSystem). The current version of this metasystem contains accumulated to this moment and formalized models, tools and methods of designing intelligent systems which are included in *OSTIS Technology*.

Remember that the main advantage of *OSTIS Technology* is flexibility of *ostis-systems*, i.e. systems which are developing on this technology. It is true also for *IMS Metasystem* because it is also *ostis-system*. It follows flexibility of *OSTIS Technology* itself, i.e. ensuring high level of development (improvement) of this technology. The main work content of *OSTIS Technology* development is reduced to creating a clearly working infrastructure which ensures organization of expertise, coordination and approval of various suggestions for improvement of *OSTIS Technology*. From formal point of view these suggestions represent suggestions for improvement of *extended knowledge base of IMS Metasystem*. Stress that at the

same time *IMS Project* aimed to developing *IMS Metasystem* and, consequently, to developing *OSTIS Technology* is an open project. Such a project allows to anyone who wishes to get into team of *OSTIS Technology* developers following all the rules of project activity organization.

IMS Metasystem interacts not only with its developers and end-users but also with other ostis-systems which are created on OSTIS Technology and represent its *child systems**. IMS Metasystem for its child systems:

- carries out automatic assembly of *child ostis-systems* starting versions on instructions which developers of these systems direct to IMS Metasystem. This way new *child ostis-systems* are generated;
- includes in *child ostis-systems* new *reusable components* from permanently replenished *OSTIS Library*. IMS makes it on its own initiative or on request of developers;
- replaces in *child ostis-systems* obsolete versions of *reusable components* by new versions from *OSTIS Library*. IMS makes it on its own initiative or on request of developers;
- includes in *child ostis-systems* a subsystem of improvement of its extended knowledge base and, if necessary, a subsystem of improvement of its *integrated knowledge processing machine* and of *user interface*;
- automatically forms and directs to *child ostis-systems* various suggestions for improvement of these systems caused by new features of permanently improved *OSTIS Technology*. These suggestions should have its own expert examination, coordination and approval within a project of improvement of the corresponding *child ostis-system*.

Thus after child ostis-systems appeared its connection to *IMS Metasystem* is not interrupted and *IMS Metasystem* become a permanent participant of process of improvement of all *child ostis-systems*.

Note also that all published materials about *OSTIS Technology* in formalized type are included in *IMS Metasystem* knowledge base [3].

More detailed information about *IMS Metasystem* see in [10].

VIII. Conclusion

In the heart of ***ontology-based design*** lies development of a complex of interconnected *subject domains* and their corresponding *ontologies*.

The main advantage of ontological approach to design is essential **increasing of flexibility** of both developed systems and design activity because of clear differentiation of (1) those *design activities* which can be executed locally within the corresponding *subject domains* and therefore do not require any coordination with design actions within another subject domains and (2) those design actions which should be coordinated between different subject domains under clearly specified coordination procedure.

Flexibility and **clearness of decomposition** of designed systems *ontological models* are the basis for effective organization of *collective design activity*.

When *designing dynamic systems* the main object of design is not only structure of these systems but also activity of these systems (activity of knowledge processing machine, activity of users during system operations and for system improvement).

Universal language of knowledge representation within intelligent systems (***SC-code***) is an effective formal base for implementation of ontology-based design of both intelligent systems and any other technical systems. The main advantages of *SC-code* are unity of syntax and semantics and also unity of language and metalanguage.

Development of *formal knowledge models* is a key issue in designing intelligent systems because quality of knowledge processing machine, of user interface and other components of intelligent system directly depend on quality of ***knowledge base***. But development of formal knowledge models is a key issue also:

- when designing any complex technical systems because direct product of any design is an information ontological model which is complete enough for the following implementation (reproduction) of this technical system;
- when organizing collective development and coordination of any other scientific and technical information, for example when developing standards;
- when processing results of any scientific researches because in this case results should be well structured, clearly represented, verified and coordinated knowledge.

More precise definition of ***ontology-base design of intelligent systems*** offered in this paper comprises solutions of the following problems:

- creating ontology of design objects – *Ontology of intelligent systems*. Such ontology is nothing but general formal theory of intelligent systems based on unified *formal models of intelligent systems*;
- creating ontology of design actions – *Ontology of designing intelligent systems*;
- creating *ontological model of intelligent system for automation of designing intelligent systems*.

Further developing ***formal models of ostis-systems*** would require include in ostis-system architecture the following additional systems:

- *subsystem of users training ostis-system* which allows end-users and developers to get new knowledge and skills during interconnection with ostis-system;
- *subsystem of information security* of ostis-system;
- *subsystem of verbal interface with other systems* (including other ostis-systems);
- *subsystem of perception and primary analysis of non-verbal information* about external environment;
- *subsystem of non-verbal influence on external environment*.

Particular aspects of ***OSTIS Technology*** are considered in the following works:

- about *technology of designing knowledge bases* of ostis-systems see [14];
- about *technology of designing knowledge processing machines see* [30];

- *about technology of designing user interfaces* of ostis-systems see [17];
- about creating *interpreters of formal models of ostis-systems* on different platforms see [18].

References

[1] The First Artificial Global Intelligence conference website. [Online]. Available: http://agi-conf.org/2008/

[2] (2016, Apr.) World's Leading Software Platform for Real-Time Expert Systems. [Online]. Available: http://www.gensym.com

[3] The IMS.OSTIS website. [Online]. Available: http://www.ims.ostis.net/

[4] (2016, Nov.) Protégé. [Online]. Available: http://protege.stanford.edu/

[5] Quillian M.R. Semantic Memory // Semantic Information Processing. – MIT Press, 1968. – p. 227-268

[6] Боргест Н.М. Онтология проектирования: теоретические основы. Часть 1. Понятия и принципы / Н. М. Боргест // Учебное пособие. – Самара: изд. СГАУ, 2010. - 91с.

[7] Боргест Н.М. Научный базис онтологии проектирования / Н. М. Боргест // Онтология проектирования. – 2013. – №1. с.7-25.

[8] Гаврилова Т.Л. и др. Инженерия знаний. Модели и методы: Учебник / Т. Л. Гаврилова, Д. В. Кудрявцев, Д. И. Муромцев. - СПб.: Издательство «Лань», 2016. - 348с.

[9] Глоба Л.С., Новогрудская Р.Л. Метод формирования инженерных расчетов на порталах знаний / Л.С. Глоба, Р.Л. Новогрудская // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2016): материалы VI Междунар.научн.-техн.конф. – Мн.: БГУИР, 2016, с.57-60

[10] Голенков В.В., Гулякина Н.А. Открытый проект, направленный на создание технологии компонентного проектирования интеллектуальных систем / В.В. Голенков, Н.А. Гулякина // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы III Междунар.научн.-техн.конф. – Мн.: БГУИР, 2013, с.55-78

[11] Голенков В.В., Гулякина Н.А. Проект открытой семантической технологии компонентного проектирования интеллектуальных систем. Часть 1: Принципы создания. / В. В. Голенков, Н.А. Гулякина // Онтология проектирования. – 2014. – N 1. с.42-64

[12] Голенков В.В., Гулякина Н.А. Проект открытой семантической технологии компонентного проектирования интеллектуальных систем. Часть 2: Унифицированные модели проектирования. / В. В. Голенков, Н.А. Гулякина // Онтология проектирования. – 2014. – N 4. с.34-53

[13] Грибова В.В., Клещев А.С. и др. Платформа для разработки облачных интеллектуальных сервисов / В.В. Грибова, А.С. Клещев и др.// КИИ-2016. Труды конференции по искусственному интеллекту. Т 3. - Смоленск, 2016. – с. 24-32.

[14] Давыденко, И.Т. Семантическая модель коллективного проектирования баз знаний / И.Т. Давыденко // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2016): материалы VI Междунар.научн.-техн.конф. – Мн.: БГУИР, 2016, с.107-114

[15] Загорулько Ю.А. Технологии разработки интеллектуальных систем, основанные на интегрированной модели представления знаний / Ю. А. Загорулько // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы III Междунар.научн.-техн.конф. – Мн.: БГУИР, 2013, с.31-42.

[16] Клещев А.С. Семантические порождающие модели. Общая точка зрения на фреймы и продукции в экспертных системах // Препринт. – Владивосток: ИАПУ ДВНЦ РАН, 1986. – 39 с.

[17] Корончик, Д.Н. Семантические модели мультимодальных пользовательских интерфейсов и семантическая технология их проектирования / Д.Н. Корончик // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2012): материалы II Междунар.научн.-техн.конф. – Мн.: БГУИР, 2012, с.339-346

[18] Корончик, Д.Н. Реализация платформы для web-ориентированных систем, управляемых знаниями / Д.Н. Корончик // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2015): материалы V Междунар.научн.-техн.конф. – Мн.: БГУИР, 2015, с.89-92

[19] Лебедев, С.В. Онтологическое проектирование подсистем оценки обстановки интеллектуальных агентов / С.В. Лебедев, М.Г. Пантелеев // КИИ-2016. Труды конференции по искусственному интеллекту. Т 2. - Смоленск, 2016. - С. 353-362.

[20] Ломов П.А. Применение паттернов онтологического проектирования для создания и использования онтологий в рамках интегрированного пространства знаний / П. А. Ломов // Онтология проектирования. – 2015. – N 2. с.233-244.

[21] Мартынов В.В. Универсальный семантический код / В. В. Мартынов. - Мн.: Наука и техника, 1984. - 132с.

[22] Мордвинов В.А. Онтология моделирования и проектирования семантических информационных систем и порталов / В. А. Мордвинов // Справочное пособие. – М.: МИРЭА, 2005. - 237с.

[23] Осипов Г.С. Методы искусственного интеллекта. 2-ое издание / Г. С. Осипов. - М.: Физматлит, 2015. - 296с.

[24] Рубашкин В.Ш. Онтологическая семантика. Знания. Онтологии. Онтологические ориентированные методы информационного анализа текстов / В. Ш. Рубашкин. - М.: Физматлит, 2012. - 348с.

[25] Рыбина Г.В. Основы построения интеллектуальных систем / Г. В. Рыбина // Учебное пособие. – М.: Финансы и статистика, 2010. - 432с.

[26] Рыбина, Г.В. Некоторые аспекты интеллектуальной технологии построения динамических интегрированных экспертных систем средствами комплекса АТ-Технология / Г.В. Рыбина, Ю.М. Блохин // КИИ-2016. Труды конференции по искусственному интеллекту. Т 3. - Смоленск, 2016. - С. 194-202.

[27] Смирнов С.В. Онтологическое моделирование в ситуационном управлении / С. В. Смирнов // Онтология проектирования. – 2012. – N2. с.16-24.

[28] Смирнов С.В. Онтологии как смысловые модели / С. В. Смирнов // Онтология проектирования. – 2013. – N2. с.12-19.

[29] Федоров И.Г. Семантический анализ языков моделирования бизнес-процессов с использованием онтологии Бунде-Ванга-Вебера / И.Г. Федоров // КИИ-2016. Труды конференции по искусственному интеллекту. Т 3. - Смоленск, 2016. – с. 115-122.

[30] Шункевич Д.В. Взаимодействие асинхронных параллельных процессов обработки знаний в общей семантической памяти / Д.В. Шункевич // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2016): материалы VI Междунар.научн.-техн.конф. – Мн.: БГУИР, 2016, с.137-144

ОНТОЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Голенков В.В.

Статья посвящена уточнению понятия формальной модели интеллектуальной системы, а также рассмотрению методик и средств проектирования интеллектуальных систем. Совокупность таких моделей, методик и средств есть не что иное, как технология проектирования интеллектуальных систем. Актуальность создания таких технологий обусловлена расширением областей применения интеллектуальных систем и, как следствие, необходимостью существенного снижения трудоемкости их разработки.

Особенностью данной статьи является рассмотрение технологии проектирования интеллектуальных систем на основе онтологий, т.е. онтологическая трактовка указанной технологии. Онтологический подход к проектированию любых классов сложных систем (в том числе и интеллектуальных систем) дает возможность четко и иерархически декомпозировать процесс проектирования любой системы заданного класса на такие проектные действия, многие из которых могут выполняться параллельно и каждое из которых может быть выполнено локально, т.е. не выходя за пределы конкретных онтологий.

Таким образом, иерархической системе проектных действий ставится в соответствие иерархическая система онтологий, в рамках которой соответствующие проектные действия могут быть выполнены. Очевидно, что это существенно ускоряет проектную деятельность путем ее распараллеливания и локализации области поиска решения при выполнении каждого проектного действия.

Проблемы, которые необходимо решить для достижения поставленной цели:

- Для повышения эффективности проектирования интеллектуальных систем необходимо иметь общую (комплексную, интегрированную, целостную) технологию проектирования интеллектуальных систем, в рамках которой были бы согласованы все необходимые частные технологии, т.е. была бы гарантирована совместимость проектных решений, которые могут быть получены в рамках различных частных технологий.
  Совместимость таких проектных решений – это совместимость различных видов компонентов интеллектуальных систем, которые могут быть продуктами разработки в общем случае различных и независимых друг от друга коллективов разработчиков. В частности, должна быть гарантирована совместимость различных видов знаний, которые входят в состав базы знаний, различных моделей решения задач, используемых интеллектуальным решателем задач, совместимость различных моделей понимания внешней информации, которая поступает в интеллектуальную систему по разным каналам, в разной форме и на разных языках.
- Чтобы создать общую интегрированную технологию проектирования интеллектуальных систем, необходимо разработать общую формальную теорию интеллектуальных систем.
- Технология проектирования интеллектуальных систем должна обеспечивать снижение трудоемкости не только при первоначальной разработке интеллектуальных систем, но и в процессе постоянного их совершенствования (модернизации, реинжиниринга) во время эксплуатации.
- Необходимо, чтобы формальные модели интеллектуальных систем, которые являются продуктами (результатами) их проектирования, были максимально просты и легко понимаемы не только интерпретаторами, которые используются для их реализации на различных платформах, но и всеми разработчиками подобных моделей.
- Так как модели представления знаний и модели обработки знаний могут различаться у разных интеллектуальных систем, основой для разработки таких моделей должен быть единый универсальный принцип, "скелет который позволяет строить иерархические многоуровневые модели представления и обработки знаний любой конфигурации.
  Для моделей представления знаний необходимо иметь возможность перехода от знаний к метазнаниям, от метазнаний к метаметазнаниям и т.д., и, в частности, от описания действий (как внутренних, так и внешних) к описанию действий сколь угодно более высокого уровня.
  Для моделей обработки знаний необходимо иметь возможность перехода от агентов, способных выполнять действия одного уровня, к коллективным агентам, способным выполнять действия сколь угодно более высокого уровня.

Несмотря на то, что уже существует достаточно много современных технологий проектирования интеллектуальных систем, они решают далеко не все указанные выше проблемы. Так в последнее время наибольшее внимание уделяется инженерии знаний и технологиям онтологического инжиниринга в ущерб другим не менее важным аспектам проектирования интеллектуальных систем.

Как следствие этого, современные технологии проектирования интеллектуальных систем:

- строятся не на основе общей формальной теории интеллектуальных систем и, следовательно, недостаточно детально рассматривают интеграцию “разнородных” компонентов интеллектуальных систем (баз знаний, машин обработки знаний, пользовательских интерфейсов), а также не имеют единой универсальной формальной основы, позволяющей в рамках технологии интегрировать самые разнообразные научные и практические результаты в области искусственного интеллекта;
- не обеспечивают совместимость разрабатываемых интеллектуальных систем и их компонентов, что затрудняет организацию одновременного проектирования разных компонентов одной системы с последующей интеграцией этих компонентов, а также разработку коллективов интеллектуальных систем;
- не обеспечивают платформенную независимость проектирования интеллектуальных систем, т.е. четкое разделение процесса разработки полных формальных моделей интеллектуальных систем и процесса разработки интерпретаторов этих моделей на различных платформах;
- не имеют формализованных методик комплексного коллективного проектирования интеллектуальных систем и, в частности, не имеют четко формализованных рамок полной независимости одновременно выполняемых ветвей проектирования и точек необходимого их согласования. Следовательно, не обеспечивают сокращение трудоемкости и сроков разработки интеллектуальных систем;

- не включают в себя методик обучения инженеров интеллектуальных систем, и, следовательно, не обеспечивают повышение их квалификации при разработке и эксплуатации этих систем;
- не поддерживают собственное развитие, в том числе, путем анализа и систематизации проектного опыта.

В основе предлагаемого подхода к созданию комплексной технологии проектирования интеллектуальных систем лежат следующие принципы:

- Использование онтологического подхода к проектированию интеллектуальных систем, т.е. подхода, основанного на иерархической системе формальных онтологий.
- Разработка принципов и соответствующих онтологий для формального смыслового представления знаний любого вида, в том числе и формальных онтологий.
- Разработка Формальной онтологии интеллектуальных систем, на основе которой выполняется унификация и упрощение формальных моделей интеллектуальных систем.
- Разработка иерархической системы согласованных (совместимых) формальных онтологий для различных видов знаний и различных моделей обработки знаний. Это обеспечивает интеграцию различных видов знаний и различных моделей обработки знаний, а также независимость от платформ их интерпретации.
- Разработка Общей модели обработки знаний, которая представляет собой коллектив агентов, работающих над общей семантической памятью, взаимодействующих через эту память и управляемых знаниями, которые хранятся в указанной памяти. Оформление этой модели в виде онтологии, частными по отношению к которой будут различные онтологии конкретных моделей обработки знаний (индуктивных, дедуктивных, четких, нечетких и т.д.).
- Разработка иерархической системы согласованных формальных онтологий проектной деятельности, направленной не только на построение интеллектуальных систем, но и на постоянную их модификацию непосредственно в процессе эксплуатации. Такая иерархическая система формальных онтологий проектирования вместе с соответствующими инструментальными должна обеспечить высокие темпы совершенствования интеллектуальных систем во время их эксплуатации.
- Обеспечение высокого уровня гибкости предлагаемой технологии благодаря тому, что технология реализуется в виде интеллектуальной метасистемы, которая обеспечивает комплексную поддержку разработки интеллектуальных систем по предлагаемой технологии и которая сама построена по этой же технологии.
- Разработка формальной онтологии совершенствования технологии проектирования интеллектуальных систем: накопление и систематизация проектного опыта, расширение библиотек многократно используемых компонентов и т.д.
- Использование методики компонентного проектирования, в основе которой лежит постоянно пополняемая библиотека многократно используемых компонентов. Это пополнение выполняют как разработчики технологии проектирования, так и разработчики конкретных интеллектуальных систем. Таким образом, предлагаемая технология, оформленная как интеллектуальная метасистема и реализованная по той же технологии, обладает высокими темпами развития, имея эффективные средства спецификации накапливаемого инженерами проектного опыта, эффективные средства спецификации новых научных результатов (т.е. принципиально новых моделей, средств и методов, предлагаемых для разработки интеллектуальных систем) и эффективные средства для внесения изменений в те модели, средства и методы, которые используются в текущий момент.
- Наличие единого фундамента, позволяющего на его основе строить различные уровни иерархии компонентов интеллектуальных систем, т.е. переходить от уровня к метауровню, от знания – к метазнанию, от действий, классов действий и способов их выполнения – к метадействиям, классам метадействий и способам их выполнения, от систем – к метасистемам. Благодаря этому появляется возможность создавать многоуровневые библиотеки многократно используемых совместимых компонентов. Все это дает возможность существенно повысить уровень компонентного проектирования, когда компьютерные системы собираются из компонентов любого уровня иерархии.
- Предлагаемая технология представляет собой открытую семантическую технологию компонентной платформенно-независимой разработки гибких совместимых интеллектуальных систем и названа нами Технологией OSTIS (Open Semantic Technology for Intelligent Systems).

В основе онтологического проектирования любых систем лежит разработка целого комплекса взаимосвязанных предметных областей и соответствующих им онтологий.

Основное достоинство онтологического подхода к проектированию – это существенное повышение гибкости как самих разрабатываемых систем, так и самой проектной деятельности благодаря четкому разделению (1) тех проектных действий, которые могут выполняться локально в рамках соответствующих предметных областей и не требовать никакого согласования с проектными действиями в других предметных областях и (2) тех проектных действий, которые должны быть согласованы между разными предметными областями, но процедура согласования которых четко определена.

Гибкость и четкость декомпозиции онтологических моделей проектируемых систем является основой для эффективной организации коллективной проектной деятельности.