and comprehensive support for the subsequent stages of the life cycle of intelligent computer systems of a new generation, in particular, it is necessary:

to unify the formalization of various models for representing various types of used information stored in the memory of intelligent computer systems and various models for solving intelligent problems to ensure semantic compatibility and simple automated integrability of various knowledge types and problem-solving models in intelligent computer systems. To do this, it is proposed to develop a basic universal abstract model for the representation and processing of knowledge, which ensures the implementation of various problem-solving models.

Let us consider the stages of solving these problems in the aspect of integrating logical models of representation and knowledge processing. The need for this consideration is caused by the need of applying logical problemsolving models in intelligent computer systems of a new generation (including knowledge-driven systems), while ensuring the quality of knowledge in accordance with the problems of knowledge management.

II .KNOWLEDGE INTEGRATION AND SEMANTIC SPACE MODELS

In order to solve the problem of unifying the for malization of various models for representing various types of information used, a model of a unified semantic representation of knowledge [3] has been developed, as well as models for representing data in the form of texts of generalized formal languages [4] and processing generalized strings (and lists) for knowledge driven systems [5]. Based on and in accordance with the model of unified knowledge representation, a family of sc-languages [1], [3], [6] has been developed, to clarify the semantics of which a model of event (distensible) sets [4] has been developed and an ontological model of spatio-temporal relations of events and phenomena for knowledge processing operations has been proposed.

In order to ensure the integration of knowledge and quality assurance in the process of knowledge integration, models for the specification and integration of knowledge are proposed. Solving the problem of knowledge integration allows considering and studying by formal means the semantic neighborhoods of sc-language text elements, the key elements of sc-languages, and studying the similarity of structures that are formed as a result of integration. Based on the proposed models, a meta-model of the semantic space was developed [4], within which it is possible to study the semantic space [7]–[14] and consider semantic subspaces of various types.

The system of transitions from texts of sc-languages to topological space is studied. Below is a fragment of the ontology that describes the types of topological spaces and inclusion relations of topological spaces of various types [?], [15].

generalized sc-tuple
=: [non-empty sc-set]

generalized sc-relation
=: [sc-set of non-empty sc-sets]
⇒ *explanation*:*
[A generalized sc-relation is a sc-set of generalized sc-tuples.]

binary sc-relatio
⇒ *explanation*:*
[A binary sc-relation is an sc-set of sc-pairs (or generalized sc-tuples to which there are two different memberships of sc-elements or the same sc-element).]

nodal sc-pair
⇒ *explanation*:*
[A nodal sc-pair is an sc-pair that cannot be denoted by a membership sc-arc (positive, negative, or fuzzy).]

slot sc-relation
⇒ *explanation*:*
[A slot sc-relation is a binary sc-relation (an sc-set of (oriented) sc-pairs) whose elements are not nodal sc-pairs.]

nodal sc-pair
⇒ *explanation*:*
[A nodal sc-pair is an sc-pair that cannot be denoted by a membership sc-arc (positive, negative, or fuzzy).]

membership phenomenon
⇒ *explanation*:*
[A membership phenomenon is a set of phenomena each of which is a slot sc-relation, while any sc-arc of permanent non-membership does not belong permanently to each of them.]

becoming*
⇒ *explanation*:*
[becoming* is a binary sc-relation between events (states) or phenomena.]

immediately before'
⇒ *first domain**
*becoming**
⇒ *second domain**
established event or phenomenon

immediately after'
⇒ *first domain**
*becoming**
⇒ *second domain**
constitutive event or phenomenon

continuance*
⇒ *explanation*:*
[The continuance* is the transitive closure of the sc-relation of becoming.]

earlier'
⇒ *first domain**
*continuance**
⇒ *second domain**

early event or phenomenon

later'

⇒ *first domain**

*continuance**

⇒ *second domain**

later event or phenomenon

sc-structure*

⇒ *explanation**:

[A sc-structure* is an sc-set that contains a nonempty support sc-subset (the set of primary elements of the sc-structure*).]

support of sc-structure'

⇒ *first domain**

*sc-structure**

⇒ *second domain**

non-empty sc-set

sc-structure'

⇒ *first domain**

*sc-structure**

⇒ *second domain**

non-empty sc-set

elementarily represented sc-set

=: [elementarily represented element']

⇒ *explanation**:

[An elementarily represented element ' is an element of an sc-structure* and an sc-set all of whose elements are elements of an sc-structure*.]

full-connectivily represented sc-set sc-set

=: [full-connectivily represented element']

⇒ *explanation**:

[A full-connectivily represented sc-set element ' is an element of an sc-structure* an sc-set all of whose elements and all its memberships are elements of an sc-structure* or an sc-arc that is an elementarily-represented element 'of this sc-structure*.]

fully represented sc-set

=: [fully represented element']

⇒ *explanation**:

[A fully represented element ' is a full-connectively represented element 'of an sc-structure* with any its element that is not an sc-arc outgoing from it connected by a membership sc-arc or a non-membership sc-arc belonging to this sc-structure*.]

sc-tuple

⇒ *explanation**:

[A sc-tuple ' is a sc-tuple ' is a full-connectively represented element 'of sc-structure* that is an sc-tuple and belongs to the sc-relation 'of this sc-structure*.]

sc-relation

⇒ *explanation**:

[A sc-relation ' is a full-connectively represented element 'of sc-structure* being a sc-relation whose elements are all sc-tuples'of this sc-structure*.].]

sc-class'

⇒ *explanation**:

[An entitive closure* is the smallest superset* (structure*) in which each element is elementarily represented'.]

entitive closure*

⇒ *explanation**:

[A sc-class' is a full-connectively represented element 'of sc-structure* all of whose elements are members of an sc-structure* that is neither an sc-relation 'nor an sc-tuple 'of that sc-struct*.]

entitive closure'

⇒ *first domain**

*entitive closure**

⇒ *second domain**

entitive closure

support of entitive closure '

⇒ *first domain**

*entitive closure**

⇒ *second domain**

non-empty sc-set

substantial closure*

⇒ *explanation**:

[A substantial closure* is the smallest superset* (structure*) in which each element is a full connectively represented element ']

substantial closure '

⇒ *first domain**

*substantial closure**

⇒ *second domain**

substantial closure

support of substantial closure '

⇒ *first domain**

*substantial closure**

⇒ *second domain**

non-empty sc-set

sc-relation of similarity by slot relations*

⇒ *explanation**:

[A similarity sc-relation by slot sc-relations* is a sc-relation that is reflexive by these slot relations, i.e. for any element included in the tuple of this sc-relation under one of the slot sc-relations, there is a tuple of this sc-relation in which it enters under each of these slot sc-relations.]

sc-relation of similarity by slot relations '

⇒ *first domain**

*sc-relation of similarity by slot relations**

⇒ *second domain**

sc-relation of similarity by slot relations

slot relations of similarity sc-relation '

⇒ *first domain**

*sc-relation of similarity by slot relations**

⇒ *second domain**

slot relations of similarity sc-relation

sc-relation of semantic similarity by slot relations*

⇒ *explanation*:*

[A semantic similarity sc-relation by slot relations* is a similarity sc-relation by slot relations* si and sj, in which each element under the slot sc-relation si can be converted to an element of the syntactic type of the element under the slot sc-relation sj; two incident sc-elements under the slot sc-relation si, within this sc-relation of semantic similarity correspond to the incident elements, respectively, under the slot sc-relation sj.]

sc-relation of semantic similarity by slot relations'

⇒ *first domain**

*sc-relation of semantic similarity by slot relations**

⇒ *second domain**

sc-relation of semantic similarity by slot relations

slot relations of semantic similarity sc-relation'

⇒ *first domain**

*sc-relation of semantic similarity by slot relations**

⇒ *second domain**

slot relations of semantic similarity sc-relation

connected sc-structure'

⇒ *explanation*:*

[A connected sc-structure* is a sc-structure* that is connected.]

support of connected sc-structure'

⇒ *first domain**

*connected sc-structure**

⇒ *second domain**

non-empty sc-set

semantic similarity of sc-structures*

⇒ *explanation*:*

[A semantic similarity of sc-structures* connects the sc-set of sc-structures* with the sc-structure* sc-relation of semantic similarity by slot screlations si, sj so that for each sc-structure* from the sc-set there is its an element and a tuple of this sc-relation of similarity, in which it is included under the slot sc-relation si, and under the slot sc-relation sj there is an element of the sc-structure *, also for each element of the scstructure there is a tuple of this sc-relation of similarity, in which it enters under the slot screlation sj, and under the slot sc-relation si enters an element of the sc-structure* from the sc-set.]

sc-relation of semantic similarity of sc-structures'

⇒ *first domain**

*semantic similarity of sc-structures**

⇒ *second domain**

*sc-relation of semantic similarity by slot relations**

semantic similarity of sc-structures'

⇒ *first domain**

*semantic similarity of sc-structures**

⇒ *second domain**

*sc-structure of semantic similarity of sc-structures**

sc-structure of semantic similarity of sc-structures'

⇒ *first domain**

*sc-structure of semantic similarity of sc-structures**

⇒ *second domain**

sc-structure of semantic similarity of sc-structures