of which SC-code constructions of any complexity are built, including more specific types of sc-elements (for example, new concepts). The memory that stores SC-code constructions is called semantic memory, or *sc-memory*.

Within the technology, several universal variants of visualization of *SC-code* constructions are proposed, such as *SCg-code* (graphic variant), *SCn-code* (nonlinear hypertext variant), *SCs-code* (linear string variant).

Within this article, fragments of structured texts in the SCn code [22] will often be used, which are simultaneously fragments of the source texts of the knowledge base, understandable to both human and machine. This allows making the text more structured and formalized, while maintaining its readability. The symbol ":=" in such texts indicates alternative (synonymous) names of the described entity, revealing in more detail certain of its features.

The basis of the knowledge base within the *OSTIS Technology* is a hierarchical system of subject domains and ontologies.

In order to solve the problems that have arisen in the design of intelligent systems and libraries of their reusable components, it is necessary to adhere to the general principles of the technology for intelligent computer systems design, as well as meet the following requirements:

- ensuring compatibility (integrability) of components of intelligent computer systems based on the unifying representation of these components;
- clear separation of the process of developing formal descriptions of intelligent computer systems and the process of their implementation according to this description;
- clear separation of the development of a formal description for the designed intelligent system from the development of various options for the interpretation of such formal descriptions of the systems;
- availability of an ontology for component design of intelligent computer systems, including (1) a description of component design methods, (2) a model of a *library of reusable components*, (3) a model of a *specification of reusable components*, (4) a complete *classification of reusable components*, (5) a description of means for interaction of the developed intelligent computer system with *libraries of reusable components*;
- availability of *libraries of reusable components of intelligent computer systems*, including component specifications;
- availability of means for interaction of the developed intelligent computer system with libraries of reusable components for installation of any types of components and their management in the created system. The installation of a component means not only its transportation to the system (copying sc-elements and/or downloading component files) but also the subsequent execution of auxiliary actions so that the component can operate in the system being created.

Based on this, in order to solve the problems set within this article, it is proposed to develop the following system of subject domains and corresponding ontologies:

Subject domain of reusable ostis-systems components
⇒ *private subject domain:*
Subject domain of a library of reusable ostis-systems components

Subject domain of the manager of reusable ostis-systems components

The *Subject domain of reusable ostis-systems components* describes the concept of a reusable component, the classification of components, and their general specification. This subject domain allows creating new and specifying existing components to add them to the library.

As a *reusable ostis-systems component*, a component of some ostis-system that can be used within another ostis-system is understood (see [13]). This is a component of the ostis-system that can be used in other ostis-systems **(child ostis-systems)** and contains all those and only those sc-elements that are necessary for the functioning of the component in the child ostis-system. In other words, it is a component of some **maternal ostis-system**, which can be used in some child ostis-system. To include a reusable component in some system, it must be installed in this system, that is, all the sc-elements of the component should be copied into it and, if necessary, auxiliary files, such as the source or compiled component files. *Reusable components* must have a unified specification and hierarchy to support compatibility with other components. The compatibility of *reusable components* leads the system to a new quality, to an additional extension of the set of problems to be solved when integrating components.

reusable ostis-systems component
:= [typical ostis-systems component]
:= [reused ostis-systems component]
:= [reusable OSTIS component]
:= [ostis-systems ip-component]
:= *frequently used sc-identifier*:
[reusable component]
⊂ *ostis-system component*
⊂ *sc-structure*

The requirements for reusable ostis-systems components inherit the common requirements for the design of software components and also include the following ones [23]:

- there is a technical possibility to embed a reusable component into a child ostis-system;