language of its specification are unified, but also allows discovering new functionalities for this system in cognition of itself. Thus, this approach allows full implementation of intelligent computer system properties, for example, reflexivity.

- It is impossible to design and implement intelligent computer systems on a software computer system that is not itself one. Representing the system specification in this form allows significantly increasing the level of its intelligence [14].
- There is no need to create additional tools for verification and analysis of the operation of the entire system, since the representation form of the system description language is unified with the language whose texts it stores and processes. This allows not only reducing the number of tools used in the design and implementation of the ostis-platform but also allows unifying the information stored in the ostisplatform and describing the ostis-platform with the purpose of using this information in the evolution of ostis-platform components. At the same time, the ostis-platform specification remains platformindependent, so when changing one implementation of the ostis-platform to another, an approach to describing the ostis-platform remains the same.

Software implementation of the ostis-platform
:= [Implementation of the sc-machine]
⇒ frequently used sc-identifier*:
[Software platform of ostis-systems]
:= [Basic software platform for mass creation of next generation intelligent computer systems]
:= [Our proposed software implementation of an associative semantic computer]
:= [sc-machine]
∈ specialized ostis-platform
∈ web-based implementation of the ostis-platform
:= [an option of implementing a platform for interpreting sc-models of computer systems, involving the interaction of users with the system via the Internet]
∈ multi-user ostis-platform implementation
∈ reusable ostis-systems component stored as source files
∈ non-atomic reusable ostis-systems component
∈ dependent reusable ostis-systems component
⇒ component address*:
[https://github.com/ostis-ai/sc-machine]
⇒ software system decomposition*:
{ • Implementation of memory in the ostis-platform
• Implementation of the subsystem of interaction with the external environment using languages of network interaction
• Implementation of the interpreter for sc-models of user interfaces
• Implementation of the basic set of platform-specific sc-agents and their common components
• Implementation of the manager of ostis-systems reusable components
}
⇒ component dependencies*:
{ • Implementation of memory of the ostis-platform
• Implementation of the subsystem of interaction with the external environment using languages of network interaction
• Implementation of the interpreter for sc-models of user interfaces
}

Software implementation of the ostis-platform
⇒ underlying principles*:

- The current Software implementation of the ostis-platform is web-oriented, so from this point of view, each ostis-system is a web site accessible online through the usual browser. This implementation option has an obvious advantage — access to the system is possible from anywhere in the world where the Internet is available, and no specialized software is required to work with the system. On the other hand, this implementation option allows multiple users to work with the system in parallel.
- The implementation is cross-platform and can be built from source texts on various operating systems. At the same time, the interaction between the client and server parts is organized in such a way that a web-interface can be easily replaced with a desktop or mobile interface, both universal and specialized ones.
- The current Software implementation of the ostis-platform is customized, i.e. does not include the Implementation of the SCP Language interpreter. At the current stage of development of the Software implementation of the ostis-platform, all functioning ostis-systems are platform-dependent. This problem is primarily related to the shortcomings of the chosen and implemented sc-memory access control model, which does not allow fully creating distributed collectives of sc-agents working on sc-memory.
- The Core of the platform is the Implementation of memory of the ostis-platform, which can simultaneously interact with both Implementation of the interpreter for sc-models of user interfaces and with any third-party applications according to the corresponding languages of

network interaction (network protocols). From the point of view of the overall architecture, Implementation of the interpreter for sc-models of user interfaces acts as one of many possible external components that interact with the Implementation of memory of the ostis-platform over the network. From the point of view of the overall architecture, Implementation of the interpreter for sc-models of user interfaces acts as one of many possible external components that interact with the Implementation of memory of the ostis-platform over the network. The current Implementation of the interpreter for scmodels of user interfaces of ostis-systems in the Software implementation of the ostis-platform is platform-dependent, since the interpreter of the basic SCP Language [17] is not fully implemented.

- The current Implementation of memory in the ostis-platform allows storing and representing sc-constructions that describe any sc-model of the ostis-system, external information constructions, not belonging to the SC-code, as well as providing different levels of access for processing these constructions. In the context of this Software implementation of the ostis-platform, Implementation of memory in the ostis-platform consists of such components as: Implementation of ostis-platform sc-memory, inside which scconstructions for sc-models of ostis-systems are represented, Implementation of ostis-platform file memory, inside which external information constructions are represented that do not belong to the SC-code, i.e. the contents of ostis-system internal files, but additionally describe, explain, and detail sc-constructions for sc-models of ostis-systems.
- Current Software implementation of the ostisplatform includes Implementation of the manager of reusable ostis-systems components. This is connected with the fact that the current Implementation of the manager of reusable ostis-systems component uses Implementation of memory of the ostis-platform to store and process the specification of installed components, regardless of their implementation language.

Principles underlying the Software implementation of the ostis-platform are only basic, all components included in the Software implementation of the ostis-platform have their own implementation features, as well as analogues that must be taken into account when implementing the entire ostis-platform.

IV. PRINCIPLES OF DOCUMENTING THE SOFTWARE PLATFORM OF OSTIS-SYSTEMS

Permanent reengineering of the components of the current Software implementation of the ostis-platform is provided by an open team of developers, while each component being developed is documented according to generally accepted principles.

Software implementation of the ostis-platform $\Rightarrow$ documentation principles*:

- Regardless of the implementation language of each Software implementation of the ostisplatform component, the specification of each component includes a specification directly described in the source files of the component itself, describing the programming interface of this component, as well as a specification as part of the ostis-platform knowledge base, describing in detail the implementation of this component, including the algorithms used. At the same time, duplication in the specification for the Software implementation of the ostisplatform components is strictly prohibited. So, for example, the specification, which is directly located in the source file with the implementation of the components themselves, describes the features of using the components from the point of view of an external or internal (that is, being part of the team) developer, and the specification, which is part of the sc-text for the knowledge base of the Software implementation of the ostis-platform, additionally includes features, proposed approaches to implementation, as well as the advantages and disadvantages of the components included in the composition.
- Each component of the Software implementation of the ostis-platform is described by means of the OSTIS Technology, that is, in the SCcode, the texts of which it processes and stores. Thus, it enables the platform to analyze its state and help maintain its life cycle without the participation of its developers. Software implementation of the ostis-platform acts as a full-fledged subject that is directly involved in its own development.
- Specification of the Software implementation of the ostis-platform is an sc-language, i.e. a sublanguage of the SC-code, for which the Syntax and Denotational semantics of the SCcode are specified. This sc-language can be represented as a family of more specific sclanguages that allow describing:
  - how sc-constructions are represented inside the ostis-platform sc-memory;

- how information constructions that do not belong to the SC-code are represented within the file memory of the ostis-platform;
- how different ostis-platform subsystems interact with each other;
- which methods and their corresponding agents interact with the ostis-platform sc-memory;
- how various interpreters for sc-models of ostis-systems (knowledge base, solver, interface) are represented and work;
- and so on.

This approach makes it possible to integrate descriptions of various components [18] that are part of the Software implementation of the ostisplatform without any particular obstacles, since
the entire Specification of the Software implementation of the ostis-platform is its knowledge base with a clearly defined hierarchy of subject domains and ontologies (that is, sc-languages that describe its implementation).

- Each developer of the Software implementation of the ostis-platform takes care of the permanent support of not only the state of its components but also the specification of these components. A quality of Software implementation of the ostis-platform is guaranteed by its team of developers who are able not only to understand the implementation details of the ostis-platform but also to contribute to the creation of mutually beneficial cooperation to achieve the set goals.

These principles can be used to describe any other software computer systems, including those software computer systems that are not implemented on this ostisplatform.

V. STRUCTURE OF THE SOFTWARE PLATFORM OF OSTIS-SYSTEMS

Implementation of memory in the ostis-platform
:= [Implementation of ostis-platform sc-memory and file memory]
:= [Our proposed software implementation of ostis platform sc-memory and file memory]
∈ reusable ostis-systems component stored as source files [11]
∈ non-atomic reusable ostis-systems component
∈ dependent reusable ostis-systems component
⇒ component address*:
[https://github.com/ostis-ai/sc machine/tree/main/sc-memory]
⇒ software system decomposition*:
{ • Implementation of ostis-platform sc-memory
• Implementation of ostis-platform file memory
• }
⇒ component dependencies*:
{ • GLib library of methods and data structures
• C++ Standard Library of methods and data structures
• Implementation of ostis-platform sc-memory
• Implementation of ostis-platform file memory
}

Implementation of ostis-platform sc-memory
:= [Software implementation of graphodynamic associative memory in the Software ostis-systems platform]
:= [Our proposed implementation of graphodynamic associative memory for ostis-systems]
∈ sc-memory implementation
∈ reusable ostis-systems component stored as source files
∈ atomic reusable ostis-systems component
∈ dependent reusable ostis-systems component
⇐ software model*: sc-memory
⇐ family of subsets*: sc-memory segment
:= [page of sc-memory]
⇐ family of subsets*: sc-memory element
⇒ component dependencies*:
{ • GLib library of methods anddata structures
• C++ Standard Library of methods and data structures
}
⇒ programming language used*:
•C
• C++
⇒ internal language*:
• SCin-code

In general, sc-memory can be implemented in different ways. So, for example, another version of ostis-platform sc-memory can be implemented using the software implementation of the Neo4j Platform [19]. The difference between this possible implementation of sc-memory and the current one is that the storage of graph constructions and the management of the flow of actions on them should be carried out to a greater extent by means provided by the Neo4j Platform; at the same time, the representation of graph constructions must be implemented in its own way, since it depends on the Syntax of the SC-code.