level, and most often by a team of such agents, each of which is assigned its own *method* describing the behavior of this agent, but already at a lower level. Therefore, we can say that the concept of atomicity/non-atomicity of *abstract sc-agents* is applicable within the framework of one *method description language*. In turn, we can talk about the hierarchy of *abstract sc-agents* from the point of view of the level of the language of description of the methods corresponding to such agents. In general, such a hierarchy can also have an unlimited number of levels, however, it is obvious that when lowering the level of the method description language, sooner or later we must approach the method description language, which will be interpreted by agents implemented at the level of textitostis-platform, and descending even lower - to the level of the method description language, interpreted at the hardware level. Thus, in order to ensure the platform independence of *ostis-systems*, it is advisable to allocate a method description language that would be interpreted at the level of *ostis-platform* and be the basis for the development of interpreters of higher-level languages. *Language SCP* is suggested as such a language (Semantic Code Programming), which is considered as an assembler for *associative semantic computer*[4].

With that stated, we will distinguish two variants of the classification of *abstract sc-agents*. Classification of *abstract sc-agents* on the basis of atomicity:

abstract sc-agent

⇒ *subdividing**:

{ • *non-atomic abstract sc-agent*
• *atomic abstract sc-agent*
}

Classification of *abstract sc-agents* based on the possibility of their implementation at the platform-independent level:

abstract sc-agent

⇒ *subdividing**:

{ • *abstract sc-agent, not implemented in the SCP language*
• *abstract sc-agent, implemented in the SCP language*
}

The classification of *abstract sc-agents* is discussed in more detail in [3].

IV. Principles of organization of problem solving in a distributed team of ostis-systems

Above, the principles of organizing the process of solving a problem by a team of agents within an individual ostis-system were considered. Given the necessity of solving problems in a distributed team of ostis-systems, it is advisable to talk about two types of multi-agent systems within the framework of OSTIS Technology:

- internal system of sc-agents over common sc-memory within some ostis-system;
- a distributed system of ostis-systems within the OSTIS Ecosystem [4].

In both cases, we can talk about hierarchy of agents:

- within the internal system of sc-agents, *atomic abstract sc-agents* and *non-atomic abstract sc-agents* are distinguished, in addition, there is a hierarchy of sc-agents from the point of view of the method interpretation language;
- Within the OSTIS Ecosystem, both *individual ostis-systems* and *collective ostis-systems* are distinguished, which in turn can consist of both *individual ostis-systems* and *collective ostis-systems*.

The key difference between the distributed system of ostis-systems and the internal system of -agents within the framework of *individual ostis-system* is the absence of a common memory that stores a common knowledge base for all sc-agents and acts as a medium for communication of sc-agents. In general, as a means of communication between agents within the framework of dedicated agent systems, it can be used:

- Shared unallocated (monolithic) memory, as in the case of sc-agents over sc-memory;
- Shared distributed memory. In this case, from a logical point of view, agents can assume that they are still working on a shared memory, within which the entire available knowledge base is stored, but in reality the knowledge base will be distributed among several ostis-systems and the transformations performed will have to be synchronized between these ostis-systems;
- Specialized communication channels. Obviously, when solving a problem in a distributed team of ostis-systems, there must be language and technical means that allow for the transmission of messages from one ostis-system to another.

All the listed methods of communication, depending on the class of the problem being solved, the *knowledge* and *skills* required for its solution, as well as the currently existing (available) set of ostis-systems, can be combined.

The idea of the maximum possible unification and convergence of the principles of problem solving within the framework of an individual ostis-system and a distributed team of ostis-systems is proposed as the basis for solving problems within a distributed team of ostis-systems. This approach has the following important advantage: if the general principles of problem solving do not depend on which specific set of ostis-systems is involved in solving a particular problem, then it becomes possible to easily switch from *individual ostis-system* to a distributed team of ostis-systems with its complication without the need to significantly revise the team

of agents, which are part of such an ostis-system and rethink the approach used to solve problems of a particular class. To switch from *individual ostis-system* to *collective ostis-system*, it is enough to do the following steps:

- Divide the set of classes of problems solved by this ostis-system into a family of subsets, each of which has some logical integrity, the criteria of which are generally determined by the developer. At the same time, these subsets may intersect, but when combined they must give the original set, so it is necessary to construct one of the possible *coverings** for the set of classes of problems solved by this ostis-system;
- For each of the selected subsets, it is necessary to form a set of *knowledge* and *skills* necessary to solve the problems of this set of classes. At the same time, in the general case, it may be necessary to revise the hierarchy of skills and their corresponding sc-agents, in particular, the transformation of some atomic sc-agents into non-atomic ones. Theoretically, it is impossible to avoid such a situation, but such situations can be practically eliminated at the stage of designing problem solvers of individual ostis-systems, making the hierarchy of agents sufficiently deep and matching atomic sc-agents with such classes of tasks, the division of which into subclasses from a practical point of view does not make sense.

  A similar situation may arise during the allocation of fragments of the knowledge base. In this case, it may be necessary to revise the hierarchy of *subject areas* and *ontologies* and, possibly, the allocation of new subject areas. As in the case of problem solvers, it is possible to avoid such a situation in practice if the hierarchy of subject areas is deep enough so that the allocation of more specific subject areas is practically impractical;
- Each set of knowledge and skills formed in this way becomes, respectively, the knowledge base and the problem solver of the new ostis-system, which will be able to implement only part of the functionality of the original ostis-system.

Such separation can be performed iteratively and for the resulting ostis-systems, in general, an unlimited number of times, creating at each iteration a new "generation" of ostis-systems obtained by decomposition of the original ostis-system.

therefore, the proposed idea of unifying the principles of problem solving in ostis-systems of any kind allows:

- from a practical point of view, remove the restriction on the expansion of functionality (training) not only of the *individual ostis-system*, but also of the *collective ostis-system*, therefore allowing us to constantly increase the functionality of the OSTIS Ecosystem as a whole.
- from a theoretical (architectural) point of view, we can talk about the fractal nature of not only the internal organization of ostis-systems, but also the collectives of ostis-systems, which, in turn, makes it possible to inherit other principles of building individual ostis systems in distributed collectives of ostis-systems, including, for example, the design methodology ostis-systems and their components and the corresponding means, as well as the principles of synchronization of parallel information processes corresponding to sc-agents.

The interaction of sc-agents within the framework of an individual ostis-system is based on the refined principle of the "bulletin board" in which agents interact through a common sc-memory for them . To implement the same idea in the case of a distributed collective ostis-system, it is necessary to select some sc-memory to perform this role. When solving problems in a distributed team of ostissystems, two options for organizing agent interaction are possible (which are the ostis-systems themselves):

- If the task being solved is quite complex and requires frequent access to several separate ostis-systems, then it is advisable to create a *temporary ostis-system* by combining separate ostis-systems, where all scagents that were part of the original ostis-systems become internal, and the principles of organizing their interaction are known. In this case, the costs of solving the problem are significantly reduced, but there are overhead costs for creating such *temporary ostis-systems*. Thus, it is necessary to separately develop criteria on the basis of which a decision will be made on the expediency of such an association. Note that in order to be able to save the result and the progress of solving the problem for subsequent use, it is advisable to combine ostis-systems based on one of the ostis-systems included in such an association, and not create a completely new ostis-system. At the same time, knowledge and skills from the combined systems will be copied into such a system, and these combined systems themselves may not change at all. Then, after solving the problem, it will be necessary to exclude from the original ostis-system those skills and knowledge that were needed only to solve this problem.

  It is important to note that the described integration of ostis-systems, due to the peculiarities of their architecture, is much easier than in other computer systems, since the principles of building both knowledge bases and problem solvers of ostis-systems initially assume the possibility of unlimited expansion of the knowledge and skills available in the system without the need to make changes to the already existing knowledge base and solver. Thus, the integration of two ostis-systems, subject to their semantic compatibility, is reduced to the usual set-

theoretic unification of their knowledge bases and problem solvers and the subsequent exclusion of duplicated components. Due to this, the creation of such temporary ostis-systems can be performed automatically, which makes the application of this approach to the organization of problem solving expedient in many cases.

- Another possible option assumes that the sc-memory of one of the ostis-systems that are part of the ostissystems team is selected as the medium for the interaction of sc-agents (both external and internal, the external ostis-system is also considered as a scagent from the point of view of the problem solving process). The following criteria for choosing this sc-memory are proposed:
  - If the task is solved repeatedly within the framework of some ostis-community (community of ostis-systems and their users, then, to coordinate the actions of sc-agents, the sc-memory of the corporate ostis-system for this ostis-community is selected;
  - If a team of ostis-systems is formed temporarily (on a one-time basis) to solve this problem, then the sc-memory of the ostis-system that initiated the solution of this problem is selected to coordinate the actions of sc-agents.

  The disadvantage of this option is the cost of communication between ostis-systems. If for some reason these costs are high (for example, due to the poor quality of the connection between the systems), then it is more appropriate to use the first of the proposed options.

In any of the proposed options, some specific sc-memory is eventually determined, which becomes an environment for the interaction of agents performing the task solution, according to the principles outlined in the . Then it is possible to clarify the concept of a *sc-agent* as a component of a problem solver in the context of distributed problem solving by a team of ostis-systems and consider as a sc-agent not only a component of the solver of an individual ostis-system, but also any ostis-system that is part of a permanent or temporary team of ostis-systems that solve any problems, since the principles of interaction ostis-systems in such a team completely coincide with the principles of interaction of sc-agents as part of the solver of an *individual ostis-system*.

Thus, we can talk about a fractal hierarchical structure (see [10]) of a distributed hybrid problem solver, within which two variants of the hierarchy of sc-agents are distinguished:

- Hierarchy of sc-agents from the point of view of the level of *method representation languages* in which the methods corresponding to these sc-agents are presented. Within this hierarchy, in turn, three levels can be distinguished that have important differences:
  - The ostis-platform sc-agent level, which provides interpretation of platform-independent level methods within the framework of an *individual ostissystem*, within which a hierarchy of presentation languages of ostis-platform level methods and corresponding means of their interpretation can be distinguished;
  - The level of platform-independent sc-agents within an *individual ostis-system*, within which a hierarchy of platform-independent method representation languages can be distinguished;
  - The level of distributed collectives of ostis-systems, at which it is also possible to talk about the *method representation languages* and their hierarchy, but in general, even individual methods can be physically stored distributed in different ostis-systems. For example, we can talk about the method representation language for the financial activities of large enterprises, but it is advisable to allocate sublanguages to describe the activities of departments of various categories and have separate ostis-systems for servicing each of the departments.
- Hierarchy of sc-agents in terms of atomicity/nonatomicity within a single *method representation language*. The formation of such a hierarchy may be appropriate at any level of the language of the method representation language and leads to the allocation of:
  - *atomic platform-dependent sc-agents* and *nonatomic platform-dependent sc-agents* at the ostis-platform level;
  - *atomic platform-independent sc-agents* and *nonatomic platform-independent sc-agents* at the platform-independent level within the framework of an individual ostis-system;
  - *individual ostis-systems* and *collective ostis-systems* at the level of problem solving within the OSTIS Ecosystem.

The presented hierarchy of *abstract sc-agents* and methods corresponding to *atomic abstract sc-agents* is illustrated in Figure 1. The label "M" in the figure conventionally denotes *methods*, the label "AA" and "NA" – *atomic abstract sc-agents* and *non-atomic abstract sc-agents*, respectively, solid arrows show the decomposition of *non-atomic sc-agents* into simpler ones, and dotted arrows – the relationship between *methods* and their operational semantics, that is, a*bstract sc-agents* that provide interpretation of these *methods*. As shown in the figure above, there should be a clear boundary between methods that are described at the *ostis-platform* level and methods that can be described at the platform-independent level. In addition, the upper part of the figure shows ostissystems that are agents within the OSTIS Ecosystem, and together with the ostis-system considered in more