Methodological and Practical Aspects of Developing the Unit Objects Pools in System-Object Models

Zhikharev A.G, Matorin S.I., Buzov. A.A, Kuznetsov A.V, Chekanov N.A.

Abstract: ***The present article deals with formal bases of pool extension procedures with UFO-elements, that represent unit objects in functional units calculus in the frame of system-object imitation modelling. It states there is developed basis on the problems of considering the equilibrium internal system parameters as the functions of outer imitational parameters without real object experiments, and the basis is developed of both computational sciences, and methodology. The given logic mathematical object description is used for design, analysis and evaluation of object or process execution.***

Key words: ***Unit-Function-Object system-object approach, imitation modelling, imitation, analogy, pool of elements, internal parameters of system, UFO-element, unit object, flow object.***

INTRODUCTION

The present article is a result of what we understand as the most important way of studying the environment, and that is analogy. By means of analogies one can form out the unified description of both parameters and fundamental features of any system (complexity, stability, reliability, etc.). This idea / concept forms the basis of a whole range of disciplines: mathematical modeling, programming, control theory and other theoretical and practical branches of science and technology. However, the methodology of the above mentioned disciplines in relation to the analysis of complex processes occurring in environments with unknown characteristics is incomplete. In other words, such an approach is not applicable to the study of complex objects / processes, since it does not contain dynamic procedures for their study, that is, it does not provide a dynamic interconnection of a static concepts system. Thus, the schematization of dynamic relationships of static concepts / objects with dynamic concepts / objects depends on understanding the basic idea of the model, which provides the necessary conditions for identifying trends in the reference area of dynamic complex objects as objects of functional analysis in the study of the conceptualization potential itself of the simulation models in wider disciplinary contexts [1].

At the same time, we believe that within the framework of simulation modeling by means of special simulators developing for complex technical devices operating in various environments, it is possible to develop an appropriate method for creating software and hardware complexes replacing a complex object or process in the real world with a sufficient degree of accuracy / reliability. To build an effective simulation model of any complexity and simplify the procedure for its design, we propose the use of built-in pools of ready-made components of the model of various modern software tools. Thus, ready-made elements pools allow you to build a simulation model from ready-made parts, which is naturally easier for the developer than programming the model from scratch. The method proposed below is an illustration of our approach.

BASIC CONCEPTS AND DEFINITIONS.

The system-object method of simulation is a modern technology for describing functioning systems, based on the *Unit-Function-Object* system approach. In order to formalize the procedures of simulation modeling of processes and systems, the authors developed the statements for calculating functional units [2, 3], within which the system-object model is represented as:

$$M=(L,S), \quad (1)$$

- where M is the model of the system;
- L is an array of model flow objects, its elements represent an object which is methodless and possesses only areas (2):

$$l=[r_1, r_2, \dots, r_k], \quad (2)$$

where:

- $l \in L$;
- k is a number of areas of the flow object l;
- $r_1, r_2, ..., r_k$ are the areas of flow object building up a ‘identifier-meaning’.

S is an array of unit objects of a model, its elements are described as follows (3):

$$s=[U, f, O], \quad (3)$$

where:

- U is an array of areas for interface flows description of the unit object s;
- f is a method of unit object s, describing the transformation function from the incoming interface flow objects L?, that are the incoming connections of the system, to the outgoing L!;
- O is an array of areas for object parameters description of the unit object (of the system) s.

Revised Manuscript Received on April 12, 2019.

ZHIKHAREV A.G., Belgorod State University, Russia.
MATORIN S.I., Belgorod State University, Russia.
BUZOV A.A., Belgorod State University, Russia.
KUZNETSOV A.V., Belgorod State University, Russia.
CHEKANOV N.A. Belgorod State University, Russia.

Moreover, the unit objects of the model M represent the key elements of the model, and the set of flow objects defines the relationship between the unit objects of the model [4].

RESULTS & DISCUSSIONS

The pool of ready-made elements of the system-object model in this case will have the form (1), and | L | = 0. That is, the pool model will have only unit objects, and will not contain flow objects. Then the pool of the system-object model [5] can be considered as a set of unit objects of the following type:

$$S'=[s_1,s_2,\ldots,s_n], \quad (4)$$

where n in the number of unit objects, stored in a pool.

Let us consider in more detail the element of the pool, or rather its formal aspect [6, 7]. As it is stated above, the pool element is a separate modelled system [8]. In the framework of calculating functional objects, the system described by expression (3) is represented as the following expression:

$$s_n=[L?, L!; f(L?)L!; O?, O!, Of] \quad (5)$$

Graphical formalism, which is an element of the pool, is stated in the following form:

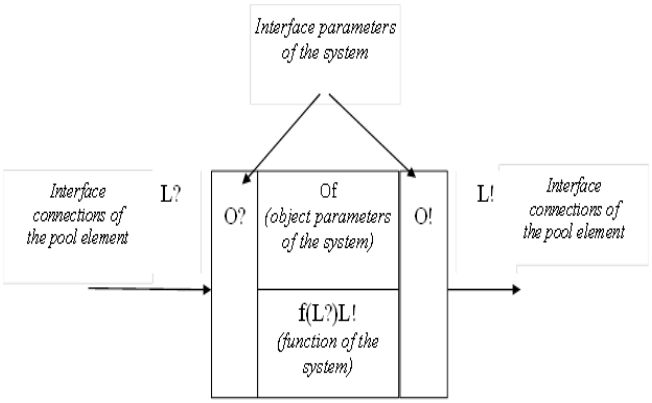

Figure 1. Graphical formalism of the pool element

As it can be seen from Figure 1, each element of the pool is a UFO-element with the corresponding interface connections, which are used to analyze the compliance of the current element with the specified characteristics. Accordingly, the pool of elements represents a set of unit objects that are not related to each other [9].

Further we consider an abstract pool $S_{M,}$ containing further elements, as Figure 2 shows.

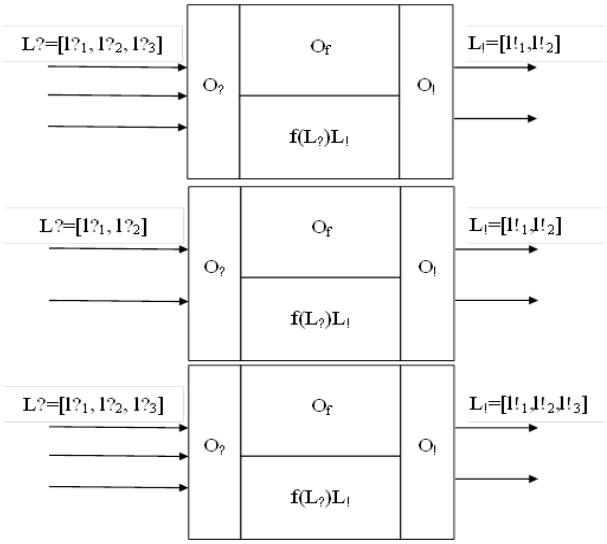

Figure 2. Graphical formalism of the abstract pool of unit objects

Obviously, for pool extending and its further use, it is necessary to consider at least two operations on system-object models: adding a unit object to the pool or exporting an item and importing a unit object from the pool. To describe these operations, an exhaustive set of parameters are the interfaces of imported and exported unit objects. The internal organization of such objects (functional and system object) does not matter for the operations in question. From Figure 2 it can be seen that for each individual unit object's interface is its identifier, that is, the name and sets of input and output stream objects with regard to their structures and area types. Further we consider a unit object with the structure of interface links, as shown in Figure 2. The formal view of this unit object is presented below:

$$s_n=[\ L?=\{l?_1,\ l?_2,\ l?_3\},\ L!=\{l!_1,l!_2\};\ f(L?)L!;\ O?,\ O!,\ Of] \quad (6)$$

Then, the interface of the unit object corresponds to the structural characteristic of the system U from expression 3, which corresponds to the main provisions of the "Unit-Function-Object" methodology [10, 11]. However, in addition to the structural component of the unit object interface, in the case of importing and exporting elements, the typical structure of interface flow objects has an important role. It is necessary to take into account the data types of the flow object areas that make up the interface part of the unit object. Thus, if for the unit object in expression 5, the interface flow objects have the following structure:

- $l?_1=(r_1, r_2)$
- $l?_2=(r_1)$
- $l?_3=(r_1)$
- $l!_1=(r_1, r_2, r_3)$
- $l!_2=(r_1, r_2)$,

then the interface of the unit object is described as follows:

$$U_s = \begin{cases} L? = \begin{cases} l?_1 = (r_1, r_2) \\ l?_2 = (r_1) \\ l?_3 = (r_1) \end{cases} \\ L! = \begin{cases} l!_1 = (r_1, r_2, r_3) \\ l!_2 = (r_1, r_2) \end{cases} \end{cases} \quad (7)$$

Generally, expression 7 can be described in the following form:

$$U_s = \begin{cases} L? = \begin{cases} l?_1 = (r_1, \dots, r_{i_1}) \\ \dots \\ l?_n = (r_1, \dots, r_{i_n}) \end{cases} \\ L! = \begin{cases} l!_1 = (r_1, \dots, r_{j_1}) \\ \dots \\ l!_m = (r_1, \dots, r_{j_m}) \end{cases} \end{cases} \quad (8)$$

Further we consider closely the operation of importing the unit object s of the system-object model M into the pool S'. Let a hierarchy of model stream objects be given containing three real stream objects with their own areas of the following form:

$$L_M = [l_1^v = \{r_1^1, r_2^1\}, l_2^v = \{r_1^2, r_2^2, r_3^2\}, l_3^v = \{r_1^3\}] \quad (9)$$

An array of flow object of the model described in expression 9 is stated in the following form:

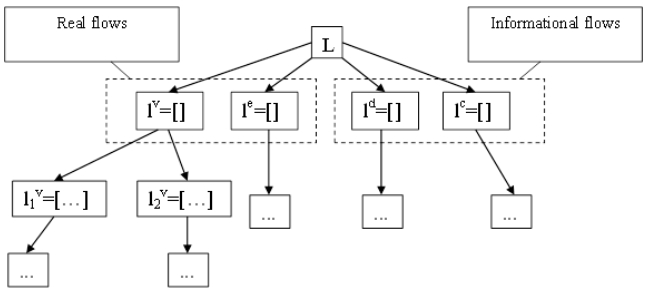

Figure 3. An Array in Flow objects of the system presented in hierarchy.

Also, let the corresponding system-object model M = (L, S) be given, then the set of unit objects has the following form:

S=[s_1=(L?=∅, L!={$l!_1$}; f(L?)L!; O?, O!, Of),

s_2=(L?={$l?_1$, $l?_3$}, L!={$l!_2$}; f(L?)L!; O?, O!, Of),

s_3=(L?={$l?_2$}, L!={$l!_3$}; f(L?)L!; O?, O!, Of)] (10)

An array of the flow objects has the following form:

L=[l_1={s_1,s_2}, l_2={s_2,s_3}, l_3={s_3,s_2}] (11)

Graphically the example described in expressions 9, 1 and 11 is stated as follows:

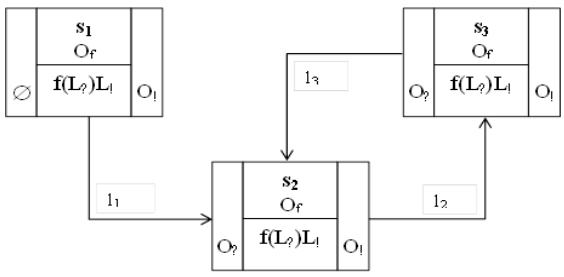

Figure 4. An example of the system-object model

Then, in order to import the unit object s2 of the system-object model M into the pool L_M we describe the operator in the following form:

L_M^* = import(M,s_2,L_M)→(L^*=∅; S^*=[s_1=(L?=∅, L!={$l!_1$}; f(L?)L!; O?, O!, Of),s_3=(L?={$l?_2$}, L!={$l!_3$}; f(L?)L!; O?, O!, Of)]; L_M^*=[s_2=(L?={$l?_1$, $l?_3$}, L!={$l!_2$}; f(L?)L!; O?, O!, Of)].

As a result of this operation we get the pool L_M^*, which is extended by the unit object s_2 and there is a system-object model $M^*(\varnothing, S^*)$ stated as follows:

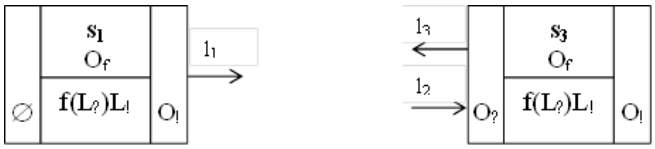

Figure 5. The result of importing a unit object into the pool

As noted above, to work with a uni object placed in the pool, you must also consider its interface. For the example in question, the interface of the s2 object is as follows:

$$U_{s_2} = \begin{cases} L? = \begin{cases} l?_1 = (r_1^1, r_2^1) \\ l?_3 = (r_1^3) \end{cases} \\ L! = \{l!_2 = (r_1^2, r_2^2, r_3^2)\} \end{cases} \quad (12)$$

It should be noted that the described operation of importing a unit object in the example involves extracting the above-mentioned element from the model into the pool. At the same time, in the course of modeling, the user can save a unit object to the pool by copying, that is, without deleting the first one from the original model and freeing the corresponding stream objects. In this case, the original model from which the item is exported to the library remains unchanged, as shown in Figure 4.

Next, we consider the operation of exporting the unit object s_2 from the pool L_M^* into the system-object model $M^*(\varnothing, S^*)$. We formulate the description of the export operator in the following form:

M^* = export(M,s_2,L_M^*)→(L^*=∅; S^*=[s_1=(L?=∅, L!={$l!_1$}; f(L?)L!; O?, O!, Of),s_2=(L?={$l?_1$, $l?_3$}, L!={$l!_2$}; f(L?)L!; O?, O!, Of),s_3=(L?={$l?_2$}, L!={$l!_3$}; f(L?)L!; O?, O!, Of)]; L_M^*=[s_2=(L?={$l?_1$, $l?_3$}, L!={$l!_2$}; f(L?)L!;O?, O!, Of)].

As it can be seen from the description of the export operation, the corresponding unit object has been added to the model, but it needs to be connected to the existing unit objects of the model. For this, we apply the operation of connecting two unit objects described in [2].

Along with this, for the selection of elements from the library, it is possible to analyze a special quantitative indicator of "measure of consistency".

To describe the algorithm for calculating the measures of consistency for a unit object with one input and one output, as shown in Figure 1, we introduce the following notation: FRFSs, which is an area or an array of the required functional states of the unit object [2], and FPSs, which is an area or an array of possible functional states [2]. Then, the elements of the given sets have the following form:

$$a_s = [A^{l_1}, A^{l_2}], \quad (5)$$

where:

A^{l_1} is the state of the input flow object l1;

A^{l_2} is the state of the output flow object l2.

The variable MOS is the desired matching coefficient for the area of possible states and the area of required states. The algorithm for calculating the measure of consistency is a sequential comparison of the elements of the FRFSs set with the elements of the FPSs set.

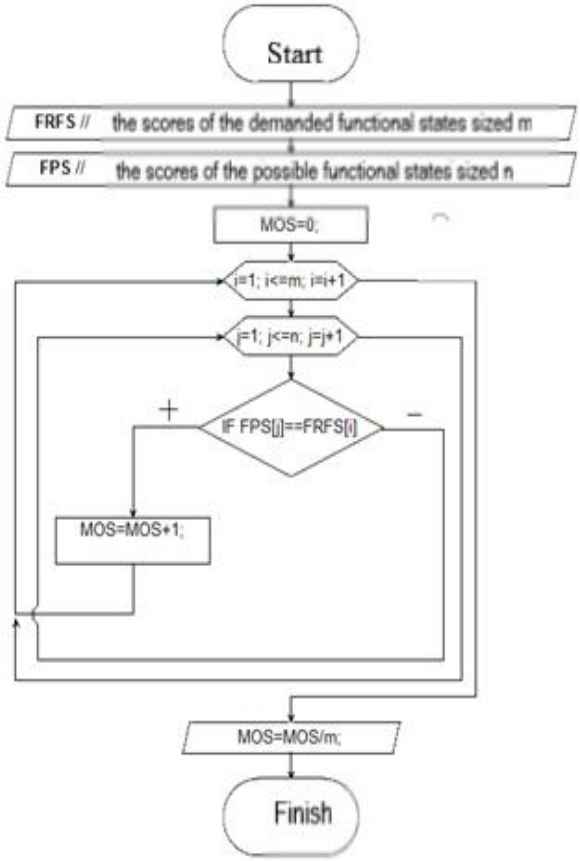

Figure 5. Algorithm for calculating the systemicity measure of the unit object

As it can be seen from Figure 5, an element of the set of required states is alternately compared with the elements of the set of possible states. If an identical state is found, then the MOS variable is incremented by one and then the transition to the new required state is carried out, since there is no need to further compare the current required state, it has already been found. Upon expiration of the external cycle, the variable MOS will contain the number of found required functional states from the set of possible states, then dividing this number by the total number of required states, we obtain the value of the systemic factor from zero to one, and the closer the coefficient lies to one, however, the more consistent system is with the supersystem request. This algorithm will work for all types of unit objects, the main problem will be the adequate formation of the set of required functional states of the unit object.

CONCLUSION

Thus, from the description of the algorithm, it follows that the presented numerical parameters of the system can be used directly to export pool elements into the model and determine the most appropriate one.

In this connection, for each component of the "Unit-Function-Object" approach, a special optimization method is applied in system-object models. The use of optimization allows making a more accurate model from the point of view of a systematic approach, a faster model from a functional point of view, as well as an object optimization in order to increase the efficiency of the processes presented in the model.

ACKNOWLEDGMENTS

The research was carried out with the financial support of the projects of the Russian Foundation for Fundamental Research №№ 18-07-00355, 19-07-00290, 19-07-00111.

REFERENCES

1. Kuznetsov, A. V. Ekzistentsial'nyy status idealizatsiy s neprostranstvennymi svoystvami kak element fizicheskoy real'nosti // Naslediye M.K. Petrova: filosofiya, kul'turologiya, naukovedeniye, regionalistika : sbornik nauchnykh statey – Belgorod : IPK BSIC, 2016. – pp. 92-95.
2. Zhikharev, A.G., Matorin, S.I., Kuznetsov, A.V., Zherebtsov, S.V., Tchekanov, N.A. To The Problem of the Coefficient Calculus of the Nodal Object in the System-Object Models // Journal of Advanced Research in Dynamical & Control Systems, Vol. 10, 10-Special Issue, 2018. – P. 1813-1817
3. Matorin, S.I., Zhikharev, A.G. Calculation of the function objects as the systems formal theory basis // Advances in Intelligent Systems and Computing 679, 2018, p. 182-191
4. Matorin, S.I., Zhikharev, A.G. Uchet zakonomernostey pri sistemno-ob'yektnom modelirovanii organizatsionnykh znaniy // Iskusstvennyy intellekt i prinyatiye resheniy, № 03 / 2018, pp. 57-68.
5. Yegorov, I.A., Matorin, S.I., Zhikharev, A.G. Sistemno-ob'yektnoye imitatsionnoye modelirovaniye khimicheskikh zagryazneniy podzemnykh vod v gornopromyshlennom klastere // Nauchnyye vedomosti Belgorodskogo gosudarstvennogo universiteta. Seriya: ekonomika, informatika, Vol. 45, № 3, pp. 510-523, 2018.
6. Zhikharev, A., Matorin, S., Egorov, I. Formal principles of system-object simulation modeling of technological and production processes // Journal of Advanced Research in Dynamical and Control Systems, 10(10 Special Issue), pp. 1806-1812, 2018
7. S.I. Matorin, A.G. Zhikharev and O.A. Zimovets Object Calculus in the System–Object Method of Knowledge Representation // Scientific and Technical Information Processing, Vol. 45, No. 5, pp. 1–10, 2018
8. Matorin, S.I., Zhikharev, A.G. Formalizatsiya sistemno-ob'yektnogo podkhoda "Unit-Function-Object" // Applied Informatics, Vol. 13, № 3(75), pp. 124-135, 2018 г.
9. Zhikharev, A.G., Matorin, S.I., Zimovets, O.A., Zhikhareva M.S., Rakov V.I. / The simulation modeling of systems taking into account their internal parameters change // Research Journal of Applied Sciences 2016. - V.11 (Issue 12). - pp. 1096-1105.
10. Matorin, S.I., Zhikharev, A.G., Zimovets, O.A. Ischisleniye ob'yektov v sistemno-ob'yektnom metode predstavleniya znaniy // Iskusstvennyy intellekt i prinyatiye resheniy. - 2017.- №3.- pp. 104-115.
11. Matorin, S.I., Zhikharev, A.G., Zimovets, O.A. Obosnovaniye vzaimosvyazey obshchesistemnykh printsipov i zakonomernostey s pozitsii sistemno-ob'yektnogo podkhoda // Trudy Instituta sistemnogo analiza. - 2017.- №3.- Vol. 67. – pp. 54-63.