On one Approach to Construction of Distributed Model of Discrete Systems

Alexander A. Sytnik
Head of the Chair of Theoretical Foundations of Informatics and Information Technologies
83, Astrahanskaya St., Saratov State University, Saratov, Russia, 410026
fax: +7(845 2) 240 446
e-mail: sytnik@sgu.sgu.ru
http://www.sgu.ru/kafedra/teorin/Sotrudnik_3.htm

Serguey V. Papshev
Assistant professor of the Chair of Theoretical Foundations of Informatics and Information Technologies
83, Astrahanskaya St., Saratov State University, Saratov, Russia, 410026
fax: +7(845 2) 240 446
e-mail: papshev@sgu.sgu.ru
http://sgu.sgu.ru/papshev/PapshevHomePage/

Abstract
This paper suggests a model of Distributed Discrete Object Environment (DDOE), for describing object-oriented computing and compares it to traditional structured automaton model.

Keywords: structured automaton, distributed computing, discrete object.

1 Introduction

The recent globalization of information processes gave rise to the standardization of methods and models, which are used for digital devices and information processes specification and design. Almost all of the suggested models are based on discrete mathematics. This paper suggests new model of global information processing. The model is developed from the ideas of the structured automata theory integrated with modern trends of distributed computing systems and object-oriented conceptions development.
Here is the essence of the idea. A structured automaton, proposed by A.M. Bogomolov and V.A. Tverdohlebov (1969, 1974), is a distributed computing system model.
The computing systems has made their way from stand-alone programs through client-server models to object-oriented computing environments such as CORBA (Common Object Request Broker: Architecture and Specification).
While client-server remote computing is possible to describe by structured automata, the object-oriented computing environments require new models.
To resolve this problem the notion of Distributed Discrete Object Environment (DDOE) is introduced in this paper and compared to traditional structured automaton model.

2 DDOE and its Characterization

Structured automaton, as distributed computing model has rigidly defined structure and description of components and their interfaces.
This model is synchronous by definition and reflects the procedural approach to computing.

2.1 What is DDOE?

The distributed discrete object environment is defined as a triple $DE = (\{O_i\}_{\in Ii}, M, L)$, where $\{O_i\}_{i \in I}$ is a set of discrete objects, M is a broker machine and L is a language used for interaction between objects and broker machine and description of its internal database.
The fig.1 shows the model of DDOE. We assume that problems of routing and addressing are solved by some low-level protocols. Therefore, DDOE represents presentation and application levels in ISO OSI model.
Discrete objects are understood as any discrete systems, which realize one or more functions. For example different automata and machines may describe them.
Inputs and outputs of these objects are connected to broker machine via virtual environment.
Broker machine stores information about currently existing objects, receives requests for some operation from objects and passes them to another object.
All these functions may be realized by multitape state machine.
Interaction between objects and broker machine is carried out via requests defined by language *L*.

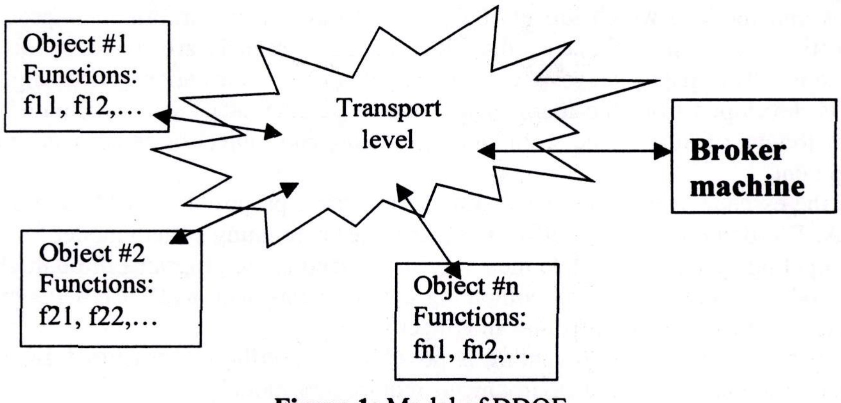

Figure 1: Model of DDOE

All statements of language L are divided into five classes:

A. request for registration of service sent by object to broker machine,
B. request for calculation of function,
C. direct request for operation execution,
D. informational message and
E. broker machine database element.

Requests of A and B types are addressed to broker machine, requests of C and D types are sent to object with a given address (it may also be a broker machine itself). The description of requests' structures follows below.

Request of A type has the following fields:

(1) service (function) name
(2) arguments (input, output)
(3) parameters
(4) object address

Request of B type has the following fields:

(1) service (function) name
(2) arguments
(3) parameters
(4) requesting object address

Request of C type has the following fields:

(1) address of object which is capable of executing the operation
(2) service (function) name
(3) arguments
(4) parameters

Request of D type has the following fields:

(1) object address
(2) message
(3) parameters

Database element of E type has the following structure:

(1) service (function) name
(2) address of object which is capable of executing the operation
(3) parameters

Broker machine is defined as a state machine $M=(S,L',\delta,)$ with four push-down tapes. $L'=L\cup\{\Lambda\}$ is a generalized alphabet (Λ is an empty symbol (word)), δ is a generalized automaton's input/output function.
S={s0,s1,s2,s3), where
s0 is an initial state,
s1 is an object function registration state,
s2 is a forward database search state,
s3 is a backward database search state.

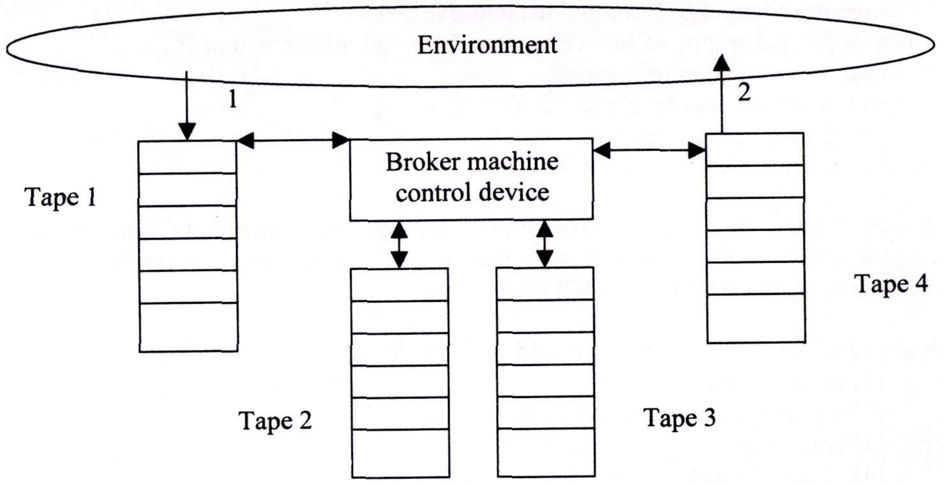

Figure 2: Structure of broker machine

The input tape 1 receives from the virtual environment via interface 1 requests of A and B types. Tapes 2 and 3 contains database elements of E type. The output tape 4 and interface 2 are used by broker machine to send messages of C and D types to the environment. The state of broker machine is determined by typified information on tapes and the state of its control device.

The function δ defines the transition of broker machine's states as follows:

δ(A,,*,*,s0)=(-,-,-,-,s1)* - *transition to the registration state,*
δ(B,,*,*,s0)=(-,-,-,-,s2)* - *transition to the database search state,*
δ(D,,*,*,s0)=(-,-,-,-,s2)* - *transition to the database search state,*

δ(A,,*,*,s1)=(Λ,E(A),-,-,s1)* - *registration of service in database due to request of A type.*

δ(B,E1,E2,,s2)=(-, -,Λ,E2,s2)* - *search on tape 2,*
δ(B,E1,E2,,s3)=(-,E2,Λ,-,s3)* - *search on tape 3,*
δ(B,,Λ,*,s3)=(Λ,-,-,-,s0)* - *search failure on tape 3,*
δ(B,E1(B),E2,,s2)=(Λ, -,E2(C(B)),C(B),s0)* - *forwarding direct request for function calculation and its registration in database due to request of B type,*
δ(B,E1,E2(B),,s3)=(Λ, -,E2(C(B)),C(B),s0)* - *dispatch of direct request for function calculation and its registration in database due to request of B type,*

δ(D,E1,E2,,s2)=(-, -,Λ,E2,s2)* - *search on tape 2,*
δ(D,E1,E2,,s3)=(-,E2,Λ,-,s3)* - *search on tape 3,*
δ(D,,Λ,*,s3)=(Λ,-,-,-,s0)* - *search failure on tape 3,*

$\delta(D(C(B)),E1(C(B)),E2,*,s2)=(\Lambda,\ -,-,D(B),s0)$ *- dispatch of message containing results of function calculation,*
$\delta(D(C(B)),E1,E2(C(B)),*,s3)=(\Lambda,\ -,-,D(B),s0)$ *- dispatch of message containing results of function calculation*

Here the asterisk symbol "*" stands for any word which is valid for the corresponding tape, "Λ" is an empty symbol (to write this symbol is to remove the top element of the stack), symbol "-" indicates no tape operation.

2.2 What Problems Does DDOE Solve?

So far the distributed discrete models were constructed in two ways. One way was to select some homogeneous components and define their interdependence (synthesis). Another way was functional decomposition of task and further modeling of elementary functions on the basis of some discrete objects (analysis).
DDOE introduces innovations, which allow us to:

- unite discrete objects of any nature in an integral interacting system;
- get access to any object included into environment;
- use only necessary properties and functions of objects;
- realize asynchronous computing models;
- easily modify environment to reflect changes in the set of objects and their functions.

2.3 DDOE vs. Structured Automaton

Structured automaton is the model of procedure-oriented distributed computing.
It is impossible to modify its components.
The only thing defined for objects in the model is interface links between them.
In DDOE outputs of the objects are not directly connected to corresponding inputs.
Instead all requests are addressed to broker machine, which decides what object can process it.
The broker machine allows to avoid direct interaction between objects. It makes the model more flexible and permits to use heterogeneous objects.
In contrast to structured automata, DDOE allows to carry out the asynchronous interaction between objects.

3 Conclusion

DDOE is likely to be a better choice then structured automaton if

- distributed system should be extended or modified;
- it is necessary to use asynchronous communication style;
- it is necessary to implement interaction between heterogeneous objects and have ability to use only part of their functions.

However, *structured automaton* is a good choice if it is necessary to construct the model of synchronous device with rigidly defined services.

References

Neumann J. V. Theory of Self-Replicating Automata, - Moscow: Mir, 1971. (University of Illinois Press, 1966).

Sytnik A.A., Posohina N.I. On Some Methods of Discrete System Behavior Simulation. Computing Anticipatory Systems: CASYS'97 - First International Conference, ed. by D.M.Debois, American Institute of Phisics, Woodbwry, New York, Conference Proceedings 437, pp. 552-559.

Bogomolov A. M., Tverdokhlebov V. A. Information Processing 1968, Noth-Holand Publishing Company, Amsterdam, 1969.

Bogomolov A. M., Tverdokhlebov V. A. Complex Systems Diagnostics. - Kiev: Naukova Dumka, 1974.

Papshev S. V., Tahirbekov Ch. B. Remote calculations in discrete mathematics. - Theoretical Foundations of Informatics and its Applications, 2, Saratov: GosUNC "College", 1998, pp. 89-93.