Modeling the Elements of an Enterprise Infocommunication System Using Colored Petri Nets

Alexey Sukonschikov, Dmitry Kochkin, Anatoly Shvetsov, Igor Andrianov, Arseny Sorokin, Svetlana Rzheutskaya
Vologda State University
Vologda, Russia
aas313@mail.ru, kochkindv@bk.ru, smithv@mail.ru, igand@mail.ru, arseny_sorokin@mail.ru, rzeyzki@yandex.ru

***Abstract*—The article is devoted to the use of colored Petri nets for modeling the structure of an enterprise infocommunication system. We have developed models of a data channel, application server and DBMS, as well as a complex model of the entire system. The article substantiates the choice of Petri nets for modeling, the results of experiments with models are presented. Comparison of simulation results and experimental data shows high adequacy of the created model.**

I. Introduction

The qualitative transition to the economics of knowledge is accompanied by significant changes including an increase in the volume of transmitted and stored data. Modern society, production and economic systems are highly complex and heterogeneous, they change and develop over time. For effective management of production and technical systems, it is necessary to have models that will adequately reflect the properties of objects, fuzziness, the presence of intelligent components, a complex hierarchy and difficult algorithms of work.

The information system of a modern enterprise or organization is a complex system that combines many related elements [1, 2]. Often a three-layer or web-architecture is chosen as the architecture of the system. It usually includes three layers (levels): clients (users), an application server, and a database management system server. A large number of information flows can pass between the levels of the system.

To analyze the information flows of the enterprise information system, it is advisable to apply modeling. With an adequate model, it is possible to carry out load testing and evaluate how well the system copes with a large load. To build such a model, it is necessary to take into account the features of the architecture of the information system, as well as the hardware configuration.

The task of modeling the enterprise information system can be divided into three components, which will be presented in the form of the following models.

1) Models of the hardware part of servers and clients. Equipment configuration is highly dependent on user needs. It is necessary to take into account the high variety and complexity of configurations – for example, the possibility of having multiple virtual servers within the same physical.

2) Models of the software part. In this paper, we focus on web architecture, which is a type of three-layer architecture. It involves the physical separation of components responsible for implementing business logic, and components responsible for working with data.

3) Models of data network. Delays occurring when data travels over the network can usually be defined as random variables with a known value of the expectation and variances.

The following methods can be considered as possible approaches to the analysis of infocommunication systems.

- Analysis of information in a real-life network by collecting statistics on servers, workstations and active network equipment.
- Testing the existing network by introducing traffic generators that simulate the flow of user data and requests to network services.
- Analysis of flows in the network based on graph flow algorithms [1], [2].
- Analysis of the workload of nodes using queuing systems [3].
- Analysis of data flows in the network and congestion of nodes based on queuing networks [4].
- Analysis of data flows in the network and congestion of nodes based on the apparatus of Petri nets of various extensions [5].

To evaluate the analysis methods and perform an reasonable choice, the following criteria were identified:

- The ability to take into account a large number of interacting elements of various nature.
- The ability to represent the studied object in the form of a system with a complex hierarchical structure and interchangeable elements (more details about modeling complex systems can be found in [6], [7]).
- Well-developed analytical apparatus, the presence of a large number of research methods.
- The ability to use the mathematical apparatus at the stage of network design.
- Low system requirements for solving the research problem.
- The ability to scale the model to a greater number of simulated elements and information flows.

ISSN 2305-7254

To perform an expert assessment, a group of experts was formed from 10 teachers and graduate students of the Department of Automation and Computer Engineering of the Vologda State University. Due to the fact that experts have different experience, level of training, and also have various scientific degrees and titles, it was proposed to introduce weight coefficients for expert opinions: graduate student – 1, candidate of sciences – 2, doctor of sciences – 4:

$$R = \{R_1, \ldots, R_n\} \mid n \in N$$

Tables with the results of evaluating mathematical apparatuses according to the proposed criteria were obtained from each expert. The final result was obtained by a weighted average score calculated by the following formula:

$$P_w = \frac{\sum_{i=1}^{n} P_i \cdot R_i}{\sum_{i=1}^{n} R_i}$$

Here P_i is the degree to which the apparatus meets the criteria given by the i-th expert; R_i is the weight of the i-th expert. The results of the expert evaluation are presented in Table I.

TABLE I. EXPERT ASSESSMENT RESULTS

Research methods	Criteria for evaluation of research methods						
	Study of a large number of elements	Representation of an object in the form of a system	Advanced analytical apparatus	Use at the design stage	Low system requirements	The ability to scale the model	Arithmetic mean of marks for all criteria
Real network information analysis	3,8	6,0	7,9	4,5	6,8	3,1	5,35
Testing an existing network	5,2	5,7	8,2	2,0	2,0	5,1	4,70
Graph flow algorithms	1,0	0,9	8,1	9,2	5,3	5,2	4,95
Queuing systems	4,4	2,6	8,1	8,1	5,2	3,3	5,28
Queuing networks	6,8	7,1	7,7	8,0	4,9	6,5	6,83
Petri nets with various extensions	7,7	7,6	7,8	7,7	5,8	7,4	7,33

The apparatus of colored Petri nets satisfies the above criteria well; therefore, it is taken as a basis in this work.

II. BASIC CONCEPTS AND DEFINITIONS

Our models of elements of the enterprise information system are based on the apparatus of colored Petri nets and created using the CPN Tools software environment [8, 9, 10, 11, 12].

The mathematical apparatus of colored Petri nets is widely used in research in the field of telecommunications: for verification of communication protocols, simulation of network devices and networks, control of production processes, etc.

The hierarchical Petri net from the conceptual point of view does not differ from the ordinary Petri net. The difference is only in the way the model is represented. To facilitate the modeling process, the concept of page is introduced. Each page contains some part of the simulated system. Pages can be linked in two ways:

1) Fusion places. This way consists in creating a set of identical places. Any change in the marking of any place implies a similar change in the markings of all positions of a given set. Pages are merged by placing identical places on different pages.

2) Substitution transitions. This method consists in the following: the transitions are defined in the net that model not elementary actions for changing markings, but some more complex processes. The net corresponding to the substitution transition is located on a special page, which is a subpage for this one. A page containing subpages is called a superpage. The connection of subpages and superpages is achieved through the use of fusion places. Each superpage place associated with a substitution transition (called a socket) corresponds to a fusion place of a subpage (called a port).

Adding fusion places and substitution transitions to the net does not change the order of operation of the atomic transitions. The net continues to operate in accordance with the rules.

Before the formal definition of the apparatus of colored Petri nets, we introduce a number of additional notation. The type of the element T is the set of all possible values of T: $T = (t_1, t_2, \ldots t_n): n \in N$. The type of the variable v or the expression e is denoted as $Type(v)$ and $Type(e)$, respectively. The set of variables in the expression e is denoted as $Var(e)$. The assignment for the set of variables V is the establishment of the correspondence of each variable $v \in V$ to some element $b(v) \in Type(v)$.

We introduce a formal definition of the set of all admissible markings of a place with a certain type. Let T be a some set (some type), M is a function that maps each element of T to a number, interpreted as a multiplicity. Then the ordered pair (T, M) is a multiset on T. Thus, the set of all multisets over the set T is the set of admissible markings of a place of type T.

A colored Petri net can be represented as follows:

$$CPN = (\sum, P, T, A, N, C, G, E, I),$$

where $\sum$ is a finite set of types, P is a finite set of places, T is a finite set of transitions, A is a finite set of arcs, N is the incidence function:

$$N : A \rightarrow P \times T \cup T \times P,$$

C is a function for determining the type of a place:

$$C : P \rightarrow \sum,$$

G is a function that establishes guard expressions for each transition, such that

$$\forall t \in T : [Type(G(t)) = B \wedge Type(Var(G(T))) \subseteq \sum],$$

E is a function that maps to each arc an expression such that

$$\forall a \in A : [Type(E(a)) = C(p)_{ms} \wedge Type(Var(E(a))) \subseteq \sum],$$

where p is the place incident to the arc, I is an initialization function (initial marking function) that maps each place to a trivial expression such that

$$\forall p \in P : [Type(I(p)) = C(p)_{ms}].$$

The main advantages of colored Petri nets for modeling network devices with quality of service are the following.

1) The ability to set complex conditions for triggering transitions and expressions on arcs to simulate the logic of the functioning of network devices.

2) The ability to use various types of data and their combination. This is important when modeling network traffic of complex structure.

This is especially important due to the complex internal structure of network devices, as it allows building more compact models.

To create submodels that are part of a complex model of an enterprise information system, the following data types, variables and functions are required (Listing 1).

Listing 1 Basic definitions for models developing

```
colset UNIT = unit;
colset INT = int;
colset TINT = INT timed;
colset INT_LIST = list INT;
colset PACKAGE = record flag:INT*dat:INT;
var i, qsize: INT;
var il: INT_LIST;
var p: PACKAGE;
var d: PACKAGE;
var dmax, dmin: INT;
fun curTime() = IntInf.toInt(!CPN'Time.model_time)
globref outfile = TextIO.stdOut;
```

The models presented in this section have been modified in order to increase usability by placing items in a separate submodel to adjust the parameters of the elements of the complex model. Centralized management is achieved by combining several places into one set.

III. Data channel model

The data channel model (Fig. 1) is responsible for introducing delays associated with data transmission. A model can repeatedly occur along the way of passing a request for modeling individual sections of a communication system. This model can be used to represent various types of networks: local, global, wireless, mobile [13], [14], [15], [16], [17].

When building a model, it becomes necessary to obtain the current value of the model time. To do this, we use the curTime() function.

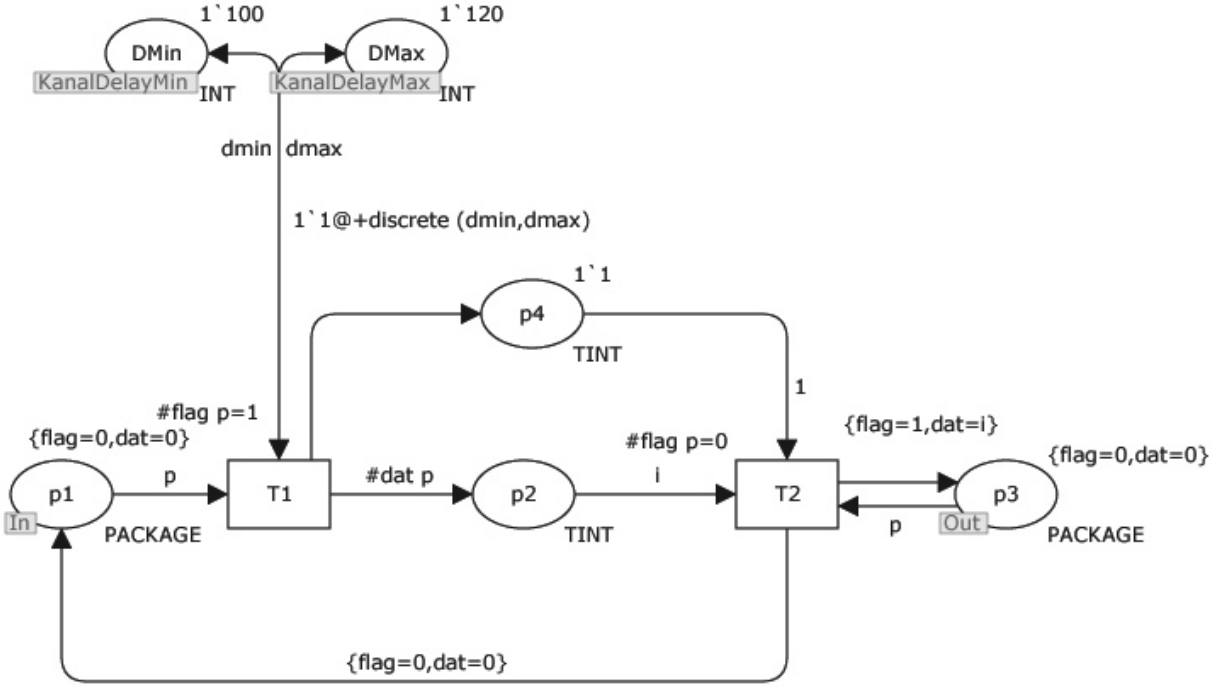

Fig. 1. Data channel model

Expressions on arcs and guard expressions for a data channel model are presented in Listing 2.

Listing 2 Basic definitions for the data channel model

```
G(T1) = #flag p=1
G(T2) = #flag p=0
E(DMin, T1) = dmin
E(DMax, T1) = dmax
E(T1, p4) = 1`1@+discrete (dmin,dmax)
E(T1, p2) = #dat p
E(T2, p3) = {flag=1,dat=i}
E(T2, p1) = {flag=0,dat=0}
```

Notice, that Fig. 1 shows a data channel model that implements simplex information transfer. To simulate duplex transmission, it is necessary to put two models in parallel, in different directions.

The elements of the model have the following functions.

In and *Out* are the input and output places, respectively. They are designed to connect various components of the model.*Chan* and *Delay* places are designed to simulate the delay of a data packet in a channel. The data packet label is put in the *Chan* place. The time delay label is placed in the *Delay* place.

Send transition moves the data packet label to the output place when the transmission time expires. The *Get* transition has the guard expression *flagp=1*. This means that the *Get* transition can work if there are labels with the *flag=1* value in the *In* place. This value indicates that the data channel is busy. When the transition is triggered, these marks will be extracted from the input place.

DMin place (a set of *ChannelDelayMin* places) and *DMax* place (a set of *ChannelDelayMax* places) are used to set the minimum and maximum packet delay times in the data channel. The *DMin* and *DMax* places are put in the submodel for controlling the data channel bandwidth.

IV. Server model

The key elements of the enterprise information system are the application server and database management system. Application server and DBMS can be launched both on one physical server and on different servers. In addition, configurations with multiple instances of the application server and a distributed DBMS are possible. Such a variety of options complicates the task of modeling and increases interest in it, since a full-scale experiment can be expensive or impossible.

The model of an application server with a DBMS operating on one physical server is presented in Fig. 2.

Expressions on arcs and guard expressions of the server model are presented in Listing 3.

Listing 3 Basic definitions for the server model

```
E(ScriptMin, SComp) = dmin
E(ScriptMax, SComp) = dmax
E(DBMin, DBComp) = dmin
E(DBMax, DBComp) = dmax
E(HWMin, HWComp) = dmin
E(HWMax, HWComp) = dmax
G(Get) = #flag p=1 andalso qsize>0
G(Drop) = #flag p=1 andalso qsize=0
E(Get, Queue) = il^^[#dat p]
E(Queue, Get) = il
E(Split, Queue) = il
E(Queue, Split) = i::il
E(Drop, In) = {flag=0,dat=0}
E(Get, QSize) = qsize-1
E(QSize, Get) = qsize
E(QSize, Split) = qsize+1
E(Split, QSize) = qsize
E(Script, SQueue) = il^^[i]
E(SQueue, Script) = il
E(SComp, SQueue) = il
E(SQueue, SComp) = i::il
E(SComp, SOQueue) = 1`i@+discrete (dmin,dmax)
E(DBQueue, DBOQueue) = 1`i@+discrete (dmin,dmax)
E(HWQueue, HWOQueue) = 1`i@+discrete
(dmin,dmax)
E(Send, Out) = {flag=1,dat=i}
```

This model can be divided into three blocks (from top to bottom): a block of places to adjust the server operation parameters, a block for receiving user requests and placing them in the service queue, a service block of user request.

The model in Fig. 2 allows to set various types of delays when processing requests to the server, which makes it possible to flexibly configure the model to represent real systems.

The last block is also divided into three parts: web application operations, database operations, work with disk. After processing the request, a data packet is sent to the user.

In the server model, *ScriptMin* (a set of *ServerScriptMin* places) and ScriptMax (a set of *ServerScriptMax* places) are places containing labels with the values of the minimum and maximum execution time of the web application program. *DBMin* (a set of *ServerDBMin* places) and *DBMax* (a set of ServerDBMax places) are places containing labels with the minimum and maximum database query execution times. *HWMin* (a set of *ServerHWMin* places) and *HWMax* (a set of *ServerHWMax* places) are places containing labels with the value of the minimum and maximum times the web application accesses files on the server's disk.

The place *In* is the input place. *Get* is a transition designed to put the user requests in the queue. *Queue* is a place simulating the request queue. The *Get* transition is triggered if the number of requests in the queue (*Queue* place) does not exceed the maximum allowed number. *QSize* (a set of *ServerQueueSize* places) is a place containing a label with the value of the remaining free space in the queue.

Drop is a transition that discards user requests when the queue overflows. *PDrop* (a set of ServerDrop places) is a place for accumulating dropped requests. *Split* is a transition that passes the request for further processing. The *Script*, *DB*, and *HW* transitions break up the user query into components.

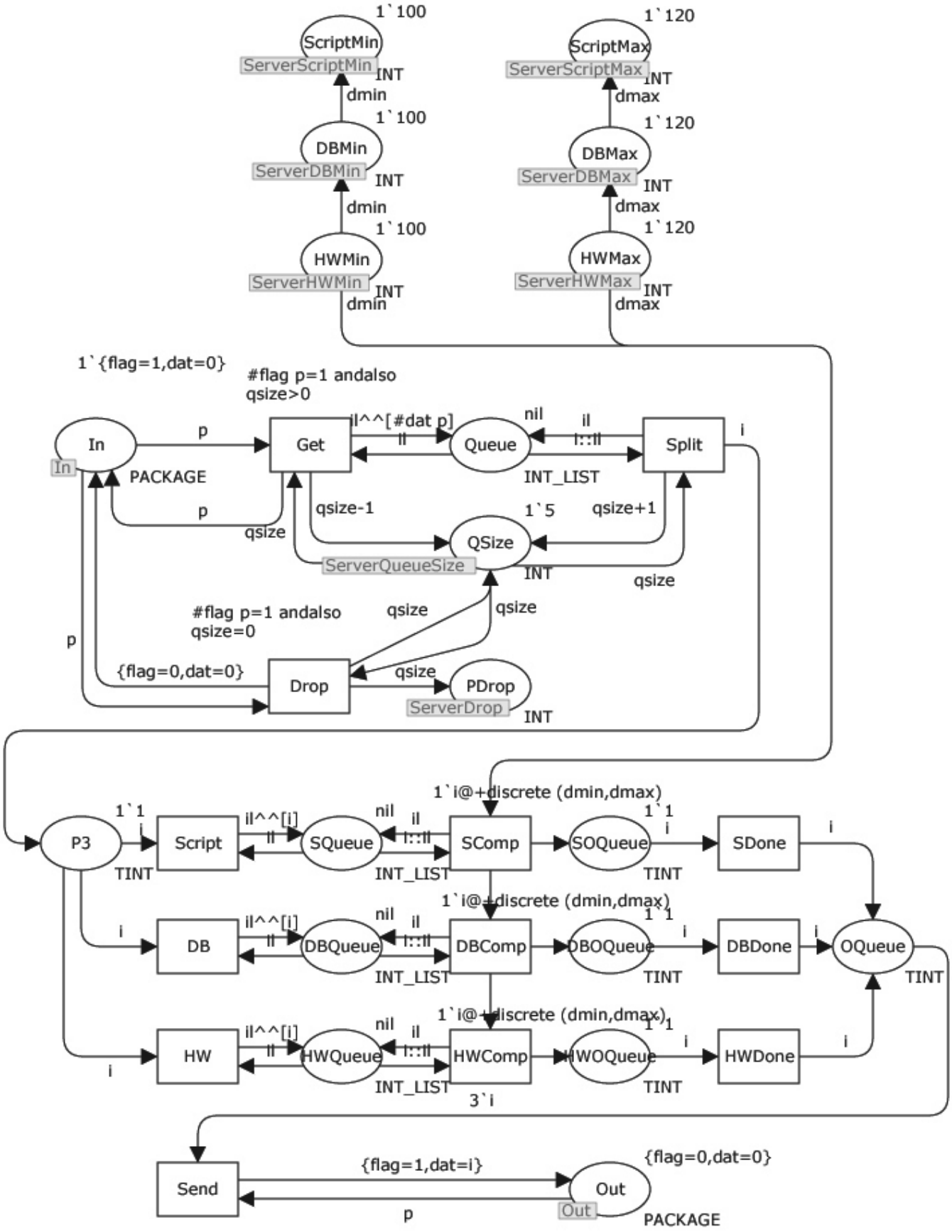

Fig. 2. Model of Application server and DBMS

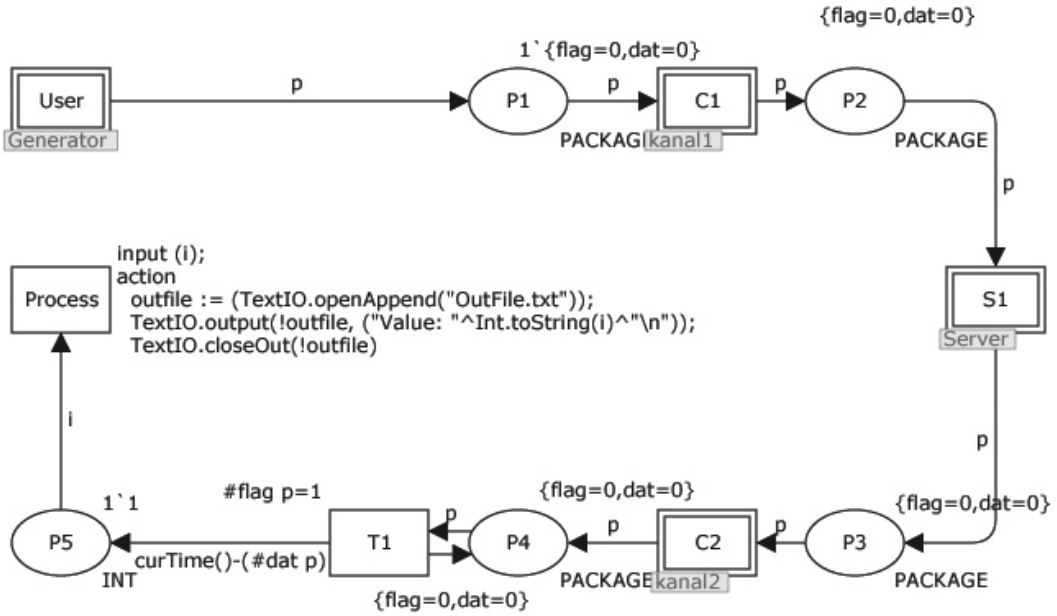

Fig. 3. Integrated Model

The place *P3* is an intermediate place when the request moves along the model.

Script models the execution of the application (script). *DB* models the execution of the query in the database. *HW* models the execution of disk operations.

The positions *SQueue*, *DBQueue* and *HWQueue* are designed for accumulating the request components.

The transitions *SComp*, *DBComp* and *HWComp* are designed for introducing a delay in the processing of the components of the request.

The positions *SOQueue*, *DBQueue* and *HWQueue* provide intermediate storage of the request.

The transitions *SDone*, *DBDone* and *HWDone* complete execution of the request and move it to the *OQueue* place. The *Send* transition sends the response to the *Out* place.

The places *ScriptMin*, *ScriptMax*, *DBMin*, *DBMax*, *HWMin*, *HWMax*, *QSize*, *PDrop* are placed in the configuration submodel for managing server parameters. The PDrop position is used to estimate the number of dropped packets.

The server model supports the ability to buffer user requests for subsequent processing. Upon reaching the maximum number of requests in the queue, all subsequent requests will be discarded.

In addition, the model provides various types of delays that arise when processing a request: execution of the application (script), querying the database, performing disk operations related to opening files.

The model can be extended to represent a multiprocessor server configuration. For this, the part of the model enclosed between the *P3* and *OQueue* places must be copied according to the number of processors (cores) in the system.

V. Web application model

In accordance with the structure of the integrated model of the enterprise information system, the models of the requests generator, data channel and server are combined into a single model for analysis (Fig. 3).

The intermediate places *P1*, *P2*, *P3*, *P4*, *P5* are designed to connect different sections of the model and submodels to each other. *User* is a submodel of the generator of requests. *C1* is a submodel of the data channel designed for modeling the delay in transmitting data to the server side. *S1* is an information system server. *C2* is a data channel submodel for simulating the delay when transmitting a server response. *T1* is a transition that accepts the user's response and places at place *P5* a label with the value of the request passing time through the model. The *Process* transition writes data about the simulation progress into a file.

VI. Experiments

To verify the adequacy of the developed models, a load test of the web server was carried out. Requests to the web server for opening various pages were formed by applications running on client machines.

The following equipment was used for testing. Firstly, the five client computers with the characteristics: Intel Core 2 DuoE4400 2GHz, DDR2 2GB, a network adapter with a speed of 100 Mbps. Secondly, the server: Intel XEON 5-2603V2 1800MHz, 1-terabyte drive, 1000 Mbit/s network adapter, 16Gb DDR3. Thirdly, the network equipment: unmanaged switch D-Link DGS-1016D with 100 and 1000 Mbps ports.

After obtaining the experimental data, a simulation was carried out. The results of simulation and experimental data are shown in tables II and III.

TABLE II. REAL AND SIMULATION RESULTS FOR STATIC WEB-PAGES

Number of requests per second	Page generation time, seconds.	
	Real	Simulation
50	0.123	0.1239
100	0.2652	0.3042
150	0.5161	0.4998
200	0.7824	0.7445
250	1.1107	0.9095
300	1.4202	1.4395
350	1.8929	1.8278
400	2.4453	2.7488
450	3.1814	3.6992
500	3.9587	3.8875

TABLE III. REAL AND SIMULATION RESULTS FOR DYNAMIC WEB-PAGES

Number of requests per second	Page generation time, seconds.	
	Real	Simulation
50	0.3139	0.3018
100	0.6709	0.7044
150	0.9752	1.0018
200	1.7101	1.4905
250	2.1191	2.421
300	3.0432	2.7423
350	4.5338	4.272
400	5.4572	5.496
450	7,6666	8,4026
500	9,5417	10,529

Load testing was carried out at different values of the flow of requests to the server. Response time was measured for an intensity of 50 to 500 requests per second in increments of 50.

Testing was carried out for two types of pages: a static page containing text and graphics, and a dynamic page. To generate a dynamic page, it was necessary to query the database. As we can see, the difference between the experimental and model data does not exceed 10%. Given how much our models are simpler than real network devices, this is a very good result.

VII. CONCLUSION

In this paper, we have confirmed that it is possible to build well-functioning models of network devices using the apparatus of colored Petri nets. The models are highly detailed, have a clear structure, allow to change individual parameters.

The models of individual network devices support composition into a single complex model through the use of special interfaces.

Comparison of real data and data obtained during modeling allows us to conclude that the presented complex model is quite adequate. So, it can be used to assess and predict some important operating parameters of the enterprise infocommunication systems.

As much as our models support the replacement of its blocks for more flexible configuration, it makes simple to adjust the models in accordance with characteristics of the simulated network. It makes them applicable for simulating a wide range of real systems.

ACKNOWLEDGEMENT

The research presented in this paper has received funding from the Russian Foundation for Basic Research, project number 19-01-00103.

REFERENCES

[1] A.V. Goldberg, S. Rao, "Length functions for flow computations", Technical report #97-055, NEC Research Institute, 1997, 18 p.
[2] R.K Ahuja, T.L. Magnanti, J.B. Orlin, "Network Flows: Theory, Algorithms and Applications", Prentice-Hall, 1993, 334 p.
[3] "System Modeling", Perm Technical University, Web: http://stratum.ac.ru/textbooks/modelir/contents.html
[4] V. Dyakonov, V. Kruglov, "MATLAB Mathematical Extension Packages. Special Reference", St. Petersburg: Piter, 2001, 480 p.
[5] K. Jensen, L. Kristensen, "Coloured Petri Nets: modeling and validation of concurrent systems", Berlin: Springer, 2009, 384 p.
[6] A.N. Shvetsov, A.A. Sukonshchiov, D.V. Kochkin, S.V. Dianov, A.N. Naimov, I.A. Andrianov, D.P. Gorbunov, "Distributed intelligent information systems and environments", Kursk: University book, 2017, 197 p.
[7] A.N. Shvetsov, A.A. Sukonschikov, D.V. Kochkin, I.A. Andrianov, "Situational intelligent decision support systems", Kursk: University Book, 2018, 251 p.
[8] K. Jensen, "Coloured Petri nets with time stamps", Aarhus: University of Aarhus, 1991.
[9] D.A Zaitsev, "Simulating telecommunication systems with CPN Tools", Odessa: ONAT, 2006, 60 p.
[10] CPN Tools Homepage, Web: http://cpntools.org.
[11] M.P. Fanti, A.M. Mangini, G. Pedroncelli, W. Ukovich, "Fleet Sizing for Electric Car Sharing Systems in Discrete Event System Frameworks", IEEE Transactions on Systems, Man, and Cybernetics: Systems, Volume: 50, Issue: 3, 2017, pp. 1161-1177.
[12] Nadia Saad Noori, Tor Inge Waag, "Application of Hierarchical Colored Petri Nets for Real-Time Condition Monitoring of Internal Blowout Prevention (IBOP) in Top Drive Assembly System", 2019 IEEE International Systems Conference (SysCon), 2019.
[13] D.V. Kochkin, "A model of a data channel based on the apparatus of colored Petri nets", in materials of the XIV All-Russian Scientific Conference "University science – to the region", Vologda State University, 2016, pp. 107-109.
[14] D.V. Kochkin, M.A. Tutynin, A.A. Sukonschikov, "Using CPN TOOLS environment for modeling colored Petri nets", in Proc. of the international scientific-methodological conference "Informatization of engineering education (INFORINO-2014)", Moscow, MPEI Publishing House, 2014, pp. 77-78.
[15] D.V. Kochkin, A.A. Sukonschikov, "Construction and analysis of a model of a segment of a TCP / IP network based on the apparatus of modified Petri nets", Management Systems and Information Technologies, vol. 4.1, 2011, pp. 144-148.
[16] Ahmad Taghinezhad-Niar, Tahmineh Javadzadeh, Leili Farzinvash, "Modeling of resource monitoring in federated cloud using Colored Petri Net", 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2017, pp. 577-582.
[17] Mohd Anuaruddin Bin Ahmadon, Shingo Yamaguchi, "On service personalization analysis for the internet of me based on PN2": 2016 IEEE International Conference on Consumer Electronics (ICCE), 2016, pp. 413-416.