Retraction Notice

The Editor-in-Chief and the publisher have retracted this article, which was submitted as part of a guest-edited special section. An investigation uncovered evidence of systematic manipulation of the publication process, including compromised peer review. The Editor and publisher no longer have confidence in the results and conclusions of the article.

ASM, AK, and NK either did not respond directly or could not be reached.

Temporary Petri-nets-based method for synthesizing network models

Amin Salih Mohammed,[a,b,*] Andriy Kovalenko,[c] and Nina Kuchuk[d]

[a]Lebanese French University, College of Engineering and Computer Science, Department of Computer Engineering, Erbil, Iraq

[b]Salahaddin University, Department of Software and Informatics Engineering, Erbil, KR-Iraq

[c]Kharkiv National University of Radio Electronics, Kharkiv, Ukraine

[d]National Technical University (KhPI), Kharkiv, Ukraine

Abstract. Many legacy systems are still essential. They run efficiently and accurately in legacy hardware and software environments. Therefore, it is necessary to migrate legacy hardware and software to modern platforms. One of the directions of migration is to service-oriented architectures. In this case, side effects are possible. Therefore, it is necessary to use the software system model. This model is used to assess the migration process of legacy software systems. An approach for building network models from the ground up is proposed in this research. Time-dependent Petri nets are used to explain the traces of events in the simulated network. A user may create a network workload model that can be utilized for network planning purposes using this technique. A method must be established to determine whether or not the created model is suitable. An example is used to determine the suitability of the Markov model for the behavior of a system with discrete states in terms of precision. An example is used to build a model for the operation of a basic software system. Providing the executable network model of the migrating system to the service identification approach input is preferred. © *2022 SPIE and IS&T* [DOI: 10.1117/1.JEI.31.6.061805]

Keywords: computing resource; composite application; hyper convergent system; time window; network modeling.

Paper 210757SS received Oct. 31, 2021; accepted for publication Mar. 22, 2022; published online Apr. 13, 2022.

1 Introduction

The issues of analyzing the quality of software packages that function in versatile multitasking environments have not been sufficiently studied. Performance analysis is important for such systems. For many systems, ensuring the required level of this indicator is necessary. This task is difficult at the software development stage. Well-tuned and validated programs in real-world conditions can show unsatisfactory performances. When writing and executing tests, considering the parallelism of computing processes is necessary. The time of arrival and processing of input data should be additional parameters for each test. This process requires unacceptably large volumes of testing and significantly complicates the interpretation of its results. One of the methods to solve the above problem is to use models of software systems. The model makes it possible to consider the specifics of the organization of parallel computing processes in modern computing systems. The model can also consider the specifics of the software implementation.

A software system model is considered one of the possible applications of the proposed approach. This model is used to assess the migration process of legacy software systems. Many of the legacy systems are still essential. They run efficiently and accurately in legacy hardware and software environments. However, such systems have a number of disadvantages. In particular, these disadvantages include high maintenance costs, scalability, and portability issues.[1] One of the directions of migration is migration to service-oriented

*Address all correspondence to Amin Salih Mohammed, amin.mohammed@lfu.edu.krd

1017-9909/2022/$28.00 © 2022 SPIE and IS&T

architecture.[2] This process depends on many factors. From the choice of migration process and service identification approach (SIA), these factors include the desired quality characteristics of the generated services.[3] In this case, side effects are possible. These side effects include system performance degradation, user resistance, and high upgrade costs.[4,5] In Ref. 5, three approaches to migration were considered. These approaches can migrate legacy systems through a top-down manner. A bottom-up strategy can be used to re-engineer its legacy software systems to a service-oriented style. A hybrid strategy can be adopted to reuse its legacy artifacts. Service identification is central to the three aforementioned migration strategies and has been recognized by practitioners as the most challenging step of the overall migration process.[6]

The services identified through SIAs must meet a range of expectations regarding their capabilities, quality of service, and efficiency of use.[1,5] For many legacy systems, the parameters and characteristics of individual modules are known. However, the use of an existing system as an input artifact for SIA is usually difficult or impossible. Therefore, providing the executable network model of the migrating system to the SIA input is preferred. The model must be adequate to the basic system.

1.1 *Analysis of Related Work*

Many models are generated by the model of external influences.[7–13] Thus, in Ref. 7, methods of nonlinear programming were used. In Ref. 8, big data processing methods were applied. The models in Refs. 9 to 11 focused on decomposition methods. In Refs. 12 and 13, the models used the mathematical apparatus of neural networks. For computer systems, a set of input tasks is called a workload.[14] Depending on the objectives of the study, the workload can be understood as input data and programs.[15] For all known methods of modeling software systems, describing the workload model is relatively difficult.[16–20] A simulation was used big data processing methods in Ref. 16. Reference 17 simulated the workload of the hypercovergent platform, and Ref. 18 simulated the cloud infrastructure. References 19 and 20 used statistical methods in workload modeling. In this case, solving a separate problem associated with the assessment of its accuracy and adequacy is necessary.[8] Widespread graph models of programs are built on the basis of studying the static structures of their source text.[21–23] However, they ignore the real dynamics of the behavior of processes. They reflect the view of the researcher or developer on the intended behavior of the system.[24] The famous route models are based on measured data from a real system. As mentioned previously, such models are excessively voluminous and insufficiently flexible.[25–27] However, all of the considered approaches ignore the design and implementation features of modern software platforms. Thus, the issues related to the analysis of the quality of software complexes operating in universal multitasking environments remain insufficiently studied.[28] This condition is true when calculating an indicator such as performance. An important task in the development of software systems is performance forecasting. A number of interesting modern approaches have been proposed in Refs. 29 to 34. However, these solutions do not take into account the specifics of SIA. A detailed analysis of existing SIAs was conducted in Ref. 5. However, the approach need is not highlighted among the considered studies. Specifically, a network model of the process of functioning of a migrating system in a new environment is fed to the SIA input.

1.2 *Goals and Structure*

The purpose of this article is to develop a method of synthesizing network models on the basis of temporary Petri nets. The developed method allows for building a model and allows a user to achieve a degree of adequacy for predicting the performance of a software package with the required reliability. The article is structured as follows. Section 2 describes the process of synthesizing a temporary Petri net using trace data. Section 3 discusses the synthesis of a temporary Petri net to model a complex of programs. Section 4 conducts an assessment of the adequacy of the network model. Section 5 present the results of a study of the effectiveness of the proposed method.

2 Synthesis of a Temporary Network Using Trace Data

A conceptual model of the functioning of the software system is used to build the model. The method is based on the representation of traces of events occurring in the system by temporary Petri nets. Random variables are used to determine the time of each transition. Each random variable is described by a general distribution law.

The concept of an ensemble of transitions is introduced. It is described by a complete group of inconsistent events. Each transition in the ensemble is determined by the probability of the corresponding event. The implementation of the ensemble of transitions allows for organizing the choice of the direction of the process development.

A method of the synthesis of temporary Petri net using trace data is proposed. This method uses ensembles of transitions and consists of the following steps.

Step 1. Highlighting N types of temporary events. These events are significant for the purposes of modeling. They make up set Ω:

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_N\}, \tag{1}$$

where ω_i is the i'th temporary event type.

The arrivals of requests for processing are considered events to build a network workload model. The processes of processing applications of specific types in accordance with specified algorithms are considered to simulate a complex of programs.

Step 2. Organization of the collection of information about event parameters from Ω; highlighting event traces.

Step 3. Highlighting M states of the software system. Sets of states S are formed on the basis of the analysis of the received traces:

$$S = \{s_1, s_2, \ldots, s_M\}, \qquad s_j = (\omega_{j_1}, \omega_{j_2}, \ldots, \omega_{j_k}), \tag{2}$$

where j_k is the number of time events defining the vector s_j and s_j is the vector describing the state j.

Step 4. Description of the route by a directed weighted graph. The set of vertices of a given graph T – highlighted states. The arcs of the graph reflect the sequence of transitions from one state to another. The weight of each graph arc is defined as the probability of the corresponding transition.

Step 5. Construction of a temporary Petri net. Network NP is built on the basis of an event graph in this manner:

$$\text{NP} = \{P, \Omega_{tr}, F, H, G_V, G_S, M_0\}, \tag{3}$$

where $P = \{p_1, p_2, \ldots, p_L\}$ is the set of event condition positions, which is determined by the presence of input arcs of the event graph; L is the number of input arcs; Ω_{tr} is the set of transitions corresponding to multiple events Ω;

$$\text{P} \cap \Omega_{tr} = \varnothing; \qquad \text{P} \cup \Omega_{tr} \neq \varnothing, \tag{4}$$

$\text{F}{:}\,\text{P} \times \Omega_{\text{tr}} \to \{0,1\}$ – Boolean function of predating multiple positions P and transitions Ω_{tr}; the function is defined by the input arcs of the event graph;

$\text{H}{:}\,\Omega_{\text{tr}} \times \text{P} \to \{0,1\}$ – Boolean function of following set transitions Ω_{tr} and items P; the function is defined by the output arcs of the event graph;

$\text{G}_\text{V}{:}\,\Omega_{\text{tr}} \times \text{R}_\text{V} \to f(Z_V)$ – matching function between set transitions Ω_{tr} and set of random variables of the execution time of events R_V; the elements R_V distributed in accordance with the distribution law Z_V; the distribution law and its parameters are determined on the basis of the trace data;

$G_S{:}\,\Omega_{tr} \times \text{P}_S \to \{0,1\}$ – correspondence function between set group transitions Ω_{tr} and the set probabilities of their triggering P_S; elements of the set P_S equal to the weights of the event graph;

$M_0{:}\,\text{P} \to \{0,1,2,\ldots\}$ – initial distribution of temporary Petri net markers. More than one marker can be found in one vertex of the network.

3 Application of Synthesis of a Temporary Petri Net in Modeling a Complex Program

An example of the application of the proposed method is used for modeling the simplest software system. A composite application package using five different data warehouse queries is considered. Requests form a set

$$\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5\}, \quad N = 5. \tag{5}$$

The following trace from the experimental execution of the package was obtained (Fig. 1).

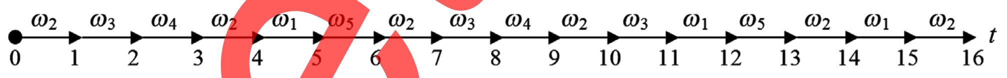
$$\text{Track} = (\omega_2, \omega_3, \omega_4, \omega_2, \omega_1, \omega_5, \omega_2, \omega_3, \omega_4, \omega_2, \omega_3, \omega_1, \omega_5, \omega_2, \omega_1, \omega_2), \quad \text{card}(\text{Track}) = 16. \tag{6}$$

A simplifying assumption in which the occurrence of a future event depends only on the present is used. This sequence of events is represented by a Markov chain. To construct this sequence, a table of precedence frequencies of elements of the set Ω (Table 1) on the basis of the trace Eq. (6) is constructed. Set $X \subset \Omega$ contains elements that determine the current state of the system. Set $Y \subset \Omega$ contains elements that define the next application to run.

The sequence of events Eq. (6) is represented by a Markov chain. In this case, the set S states of the system are isomorphic to the set Ω. The graph corresponding to trace Eq. (6) is shown in Fig. 2.

Graph arcs on Fig. 2, outgoing from vertices s_4 and s_5, are implemented by conventional time transitions. Graph arcs outgoing from vertices s_1, s_2, and s_3 are implemented by ensembles of transitions. The number of transitions in the ensemble is determined by the number of outgoing arcs. The probability of firing a transition from an ensemble is determined by the weight of the corresponding arc. In accordance with steps 1 to 5, a temporary Petri net [Eq. (3)] was constructed.

The starting marker hits the top p_2 because the event ω_2 indicates the start of processing a package of composite applications. A vertex p_0 is introduced. It corresponds to the "start of the trace" condition. Transition ω_0 generates a random package start time. This transition is the simplest workload model. It allows for the analysis of the behavior of a set of programs under various characteristics of the request flow. The probability of transitions in the Petri network corresponds to the transitions in Fig. 2. Temporary Petri net for the trace on Fig. 1 is given in Fig. 3. As a result, an ordinary active safe Petri network is formed, allowing for the investigation of closed systems. The proposed method for synthesizing a temporary Petri net using trace data for modeling a complex of programs is analyzed.



ω_2 ω_3 ω_4 ω_2 ω_1 ω_5 ω_2 ω_3 ω_4 ω_2 ω_3 ω_1 ω_5 ω_2 ω_1 ω_2
t
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Fig. 1 Event-time trace.

Table 1 Frequency of the precedence of elements of the set Ω.

	X				
Y	ω_1	ω_2	ω_3	ω_4	ω_5
ω_1	0	1	0	0	2
ω_2	2	0	3	0	0
ω_3	2	0	0	2	0
ω_4	0	2	0	0	0
ω_5	0	2	0	0	0

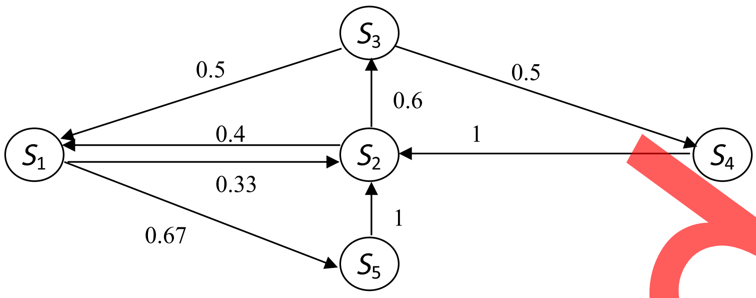


Fig. 2 Trace graph on Fig. 1.

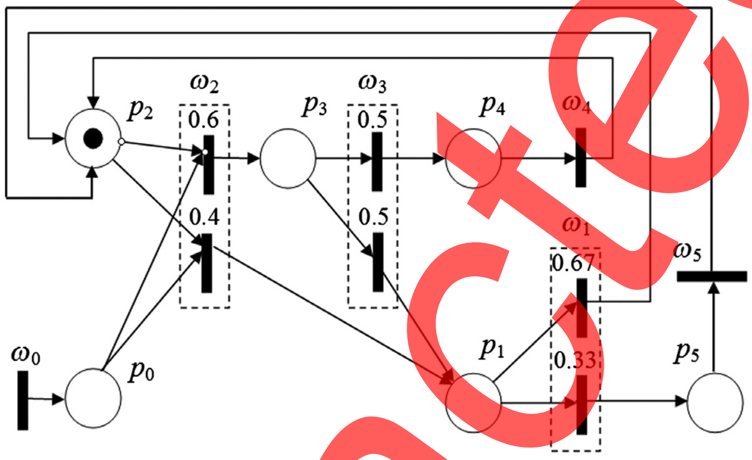


Fig. 3 Temporary Petri net for the trace on Fig. 1.

The prerequisites for applying the method are as follows:

- availability of information about the sequence and duration of network events;
- use of measuring monitors; and
- the assumption of stationarity of the probabilities of system transitions in the selected states.

The advantages of the proposed method are as follows:

- the focus on the study of parallel processes;
- the ability to use different levels of detail and hierarchical modeling, for hierarchical Petri nets can be used for this;
- the ability to study using one model of the workload and the complex of programs;
- the ability to decompose and assemble models; and
- the ease of accounting for the state of the external environment.

The main disadvantage of the method is the need for constant collection of measurement information in a computer system. The degree of model adequacy depends on the accuracy of the measurement information.

4 Assessment of the Adequacy of the Network Model

The main requirement for the model is its adequacy to the real system. The processes of functioning of real software systems cannot be described fully and in detail. This condition is primarily due to their significant complexity. Therefore, one could only talk about the degree of adequacy of the model. Improving the degree of adequacy can be achieved using different levels of detail. Choosing a criterion corresponding to the problem being solved is necessary for assessing the degree of adequacy of the model. In this paper, the adequacy assessment is based on the degree to which the behavior of the model matches the system. In other words, the adequacy of the description of the dynamics of the process under study is analyzed.

In step 3, the proposed method has a configurable parameter M—the number of system states. It depends on the length of the prehistory D, on the condition $D = 0$ (Markov process without prehistory). The hypothesis of the Markov character of the processes under study ($D = 0$) is used to provide the most compact stochastic model. Magnification of D by considering the length of the history of the entire trace gives a more accurate result. However, this process significantly increases the complexity of the tracing model. The upper limit of the possible number of states of the system is calculated as

$$M_{\max} = A_N^{D+1} = N!/(N - D - 1)!, \tag{7}$$

where A_N^{D+1} is the variations and "!" is the factorial sign.

When modeling a complex of programs, solving the problem of choosing a specific value for the duration of the prehistory of the development of processes is necessary. In this case, assessing the reliability of the representation of the behavior of the system by the trace network model is necessary.

The set of possible initial states of the system is represented as

$$S_0 = \{s_1^{(0)}, s_2^{(0)}, \dots, s_N^{(0)}\}. \tag{8}$$

S_0 is the isomorphic Ω, and $D = 0$.

The required level of model adequacy can be achieved by increasing the value D iteratively. In this case, this is the dimension of the set S_0. Therefore, the permissible level of complexity of the model must not be exceeded.

One step ξ iterative process and a set of events are considered S_ξ. For a given set, the probabilities $P_{ij}^{(n,\xi)}$ system transition from state i in state j for n steps ($n \in N$ is the natural number) was defined. Consider the matrix $Q^{(\xi)} = \|q_{ij}^{(\xi)}\|$ transition probabilities of a Markov chain, which has the dimension ρ. Then, $P_{ij}^{(0,\xi)} = q_{ij}^{(\xi)}$.

The generating function of the process is

$$\tilde{P}_{ij}^{(\xi)}(\zeta) = \sum_{n=0}^{\infty} P_{ij}^{(n,\xi)} \zeta^n, |\zeta| < 1, \tag{9}$$

where ζ is the generating function parameter.

Multiplying two sides of equality Eq. (9) to $\zeta \cdot q_{ij}^{(\xi)}$ and summing over $i = \overline{1, \rho}$, the ratio obtained is

$$s \sum_{i=1}^{\rho} q_{ij}^{(\xi)} \tilde{P}_{ij}^{(\xi)}(\zeta) = \sum_{n=0}^{\infty} \sum_{i=1}^{\rho} q_{ki} P_{ij}^{(n,\xi)} \zeta^{n+1} = \tilde{P}_{ij}^{(\xi)}(\zeta) - P_{kj}^{(0)}, \tag{10}$$

which defines the systems of equations in the form

$$\tilde{P}_{kj}^{(\xi)}(\zeta) - \zeta \sum_{i=1}^{\rho} q_{ij}^{(\xi)} \tilde{P}_{ij}^{(\xi)}(\zeta) = P_{kj}^{(0)}, k = \overline{1, \rho}, j = \overline{1, \rho}. \tag{11}$$

Their solutions for fixed k and ζ are functions of the form

$$\tilde{P}_{ij}^{(\xi)}(\zeta) = \frac{G_{ij}(\zeta)}{D(\zeta)}. \tag{12}$$

Fraction Eq. (12) decomposes into simple fractions

$$\tilde{P}_{ij}^{(\xi)}(\zeta) = \sum_{\lambda=1}^{\rho} \frac{g_{ij}^{(\lambda)}}{1-\zeta \cdot r_\lambda}, \tag{13}$$

where r_λ is the matrix eigenvalues $Q^{(\xi)}$ transition probabilities and $g_{ij}^{(\lambda)}$ is the expansion coefficients.

Let h be the arbitrary left eigenvector of a matrix $Q^{(\xi)}$, where

$$hQ^{(\xi)} = r_\lambda h. \tag{14}$$

Consider the sum of the elements of the vectors on the left and right sides of the equality Eq. (14):

$$\sum_{j=1}^{\rho}\sum_{i=1}^{\rho} h_i q_{ij}^{(\xi)} = \sum_{j=1}^{\rho} h_i \sum_{i=1}^{\rho} q_{ij}^{(\xi)} = \sum_{j=1}^{\rho} h_i = r_\lambda \sum_{j=1}^{\rho} h_i. \tag{15}$$

Provided that all h_i are positive, $r_\lambda = 1$ determines the spectral radius of the matrix $Q^{(\xi)}$. Therefore, all of its eigenvalues satisfy the inequality

$$|\, r_\lambda| \leq 1. \tag{16}$$

Therefore,

$$\frac{1}{1-\zeta \cdot r_\lambda} = \sum_{i=0}^{\infty} \zeta^n r_\lambda^n, \tag{17}$$

and from Eq. (13), it follows that

$$P_{ij}^{(n,\xi)} = \sum_{\lambda=1}^{\rho} g_{ij}^{(\lambda)} r_\lambda^n. \tag{18}$$

To find the values of the coefficients $g_{ij}^{(\lambda)}$, an iterative process is used:

$$P_{ij}^{(n+1,\xi)} = \sum_{\lambda=1}^{\rho} g_{ij}^{(\lambda)} r_\lambda^{n+1} = \sum_{k=1}^{\rho} q_{ik}^{(\xi)} P_{kj}^{(n,\xi)} = \sum_{k=1}^{\rho} q_{ik}^{(\xi)} \sum_{\lambda=1}^{\rho} g_{kj}^{(\lambda)} r_\lambda^n = \sum_{\lambda=1}^{\rho} r_\lambda^n \sum_{k=1}^{\rho} q_{ik}^{(\xi)} g_{kj}^{(\lambda)}. \tag{19}$$

From Eq. (19), the following is obtained:

$$g_{ij}^{(\lambda)} r_\lambda = \sum_{k=1}^{\rho} q_{ik}^{(\xi)} g_{kj}^{(\lambda)}. \tag{20}$$

Conversely,

$$P_{ij}^{(n+1,\xi)} = \sum_{\lambda=1}^{\rho} g_{ij}^{(\lambda)} r_\lambda^{n+1} = \sum_{k=1}^{\rho} P_{ik}^{(n,\xi)} q_{kj}^{(\xi)} = \sum_{k=1}^{\rho}\sum_{\lambda=1}^{\rho} g_{ik}^{(\lambda)} r_\lambda^n q_{kj}^{(\xi)} = \sum_{\lambda=1}^{\rho} r_\lambda^n \sum_{k=1}^{\rho} g_{ik}^{(\lambda)} q_{kj}^{(\xi)}. \tag{21}$$

Hence,

$$g_{ij}^{(\lambda)} r_\lambda = \sum_{k=1}^{\rho} g_{ik}^{(\lambda)} q_{kj}^{(\xi)}. \tag{22}$$

In matrix form, Eqs. (20) and (22) can be rewritten as

$$G^{(\lambda)}Q^{(\xi)} = r_\lambda G^{(\lambda)}, \tag{23}$$

$$Q^{(\xi)}G^{(\lambda)} = r_\lambda G^{(\lambda)}. \tag{24}$$

Thus, the columns of the matrix $G^{(\lambda)}$ are the right eigenvectors of the matrix $Q^{(\xi)}$. They are determined at $r = r_\lambda$ nonzero solutions $x_i^{(\lambda)}$ systems of equations:

$$\sum_{k=1}^{\rho} q_{kj}^{(\xi)} x_k - r \cdot x_i = 0; \quad i = \overline{1,\rho}. \tag{25}$$

Matrix rows $G^{(\lambda)}$ are left eigenvectors. They are determined by nonzero systems solutions $y_j^{(\lambda)}$

$$\sum_{k=1}^{\rho} y_k q_{kj}^{(\xi)} - r \cdot y_j = 0; \quad j = \overline{1,\rho}. \tag{26}$$

Then, Eq. (27) up to a constant factor $C^{(\lambda)}$ is defined as

$$g_{ij}^{(\lambda)} = C^{(\lambda)} x_i^{(\lambda)} y_j^{(\lambda)}. \tag{27}$$

The values $C^{(\lambda)}$ are calculated on the basis of the orthonormality of the left and right eigenvectors:

$$C^{(\lambda)} \cdot \sum_{k=1}^{\rho} x_k^{(\lambda)} y_k^{(\lambda)} = 1 \Rightarrow C^{(\lambda)} = \left(\sum_{k=1}^{\rho} x_k^{(\lambda)} y_k^{(\lambda)}\right)^{-1}. \tag{28}$$

Thus, the user can calculate the theoretical values of the transition probabilities $P_{ij}^{(n,\xi)}$.

From the data of the observed trace of the states of the real system, the frequencies of the system transition from state i in state j for $k = \overline{1,N}$ steps are represented by the matrix $V_i = \{v_{ij}^{(k)}\}$, where

$$\sum_{j=1}^{N} v_{ij}^{(k)} = n_i^{(k)}, \tag{29}$$

with $n_i^{(k)}$ being the number of observations.

Suppose the number of observations $n_i^{(k)}$ is sufficient to apply the Pearson criterion. Statistics characterizing the deviation of experimental frequencies from the corresponding theoretical values can take the value

$$\chi_n^2(V_i) = \sum_{j=1}^{N} \frac{[v_{ij}^{(k)} - n_i^{(k)} \cdot P_{ij}^{(k)}]^2}{n_i^{(k)} \cdot P_{ij}^{(k)}} = \sum_{j=1}^{N} \frac{[v_{ij}^{(k)}]^2}{n_i^{(k)} \cdot P_{ij}^{(k)}} - n_i^{(k)}. \tag{30}$$

Let us set the significance level α. Then, the hypothesis regarding the behavior of a system with the number of states L is rejected when the value exceeds $\chi_n^2(V_i)$. This value is distributed in accordance with the law χ^2 with $L-1$ degree of freedom. It corresponds to the table value $\chi^2_{1-\alpha,L-1}$.

For large values L, information criterion could be used:

$$J_C = \frac{{}_i^{(k)} - M(H_i^{(k)})}{\sqrt{D(H_i^{(k)})}}, \tag{31}$$

where

$$ {}_{i}^{(k)} = -\sum_{j=1}^{L} \frac{v_{ij}^{(k)}}{n_{i}^{(k)}} \cdot \ln\left(\frac{v_{ij}^{(k)}}{n_{i}^{(k)}}\right), \tag{32}$$

the statistical estimation of the entropy of an empirical distribution is

$$M(H_{i}^{(k)}) = h_{i}^{(k)} - (L-1)/n_{i}^{(k)}, \tag{33}$$

the expectation function is

$$D(H_{i}^{(k)}) = \frac{1}{n_{i}^{(k)}}\left(\sum_{j=1}^{L} P_{ij}^{(k)} \cdot \ln^{2}(P_{ij}^{(k)}) - [h_{i}^{(k)}]^{2}\right), \tag{34}$$

and the variance of entropy of theoretical distribution is

$$h_{i}^{(k)} = -\sum_{j=1}^{L} P_{ij}^{(k)} \cdot \ln(P_{ij}^{(k)}). \tag{35}$$

Statistics are normally distributed with zero mathematical expectation and unit variance. For a given level of significance α and distribution quantile $u_{1-\alpha}$, the equality holds:

$$|J_{C}| \leq u_{1-\alpha}. \tag{36}$$

The power information criterion is practically superior to the Pearson criterion. In this case, the probability of rejecting a correct hypothesis with a large number of observations is lower.

With a large number of system states, obtaining the expressions for theoretical probabilities $P_{ij}^{(n)}$ is difficult. In this case, as an alternative to the analytical approach, the values of theoretical probabilities can be obtained using the Monte Carlo method, that is, when simulating the behavior of the corresponding Markov chain. In this case, the problem of testing the hypothesis for the correspondence of experimental and model probability distributions is solved on the basis of a criterion of the form

$$\chi_{n}^{2}(V_{i}) = n_{i}^{(k)} m_{i}^{(k)} \sum_{(j)} \frac{(v_{ij}^{(k)}/n_{i}^{(k)} - \omega_{ij}^{(k)}/m_{i}^{(k)})^{2}}{v_{ij}^{(k)} + \omega_{ij}^{(k)}}, \tag{37}$$

where $\omega_{ij}^{(k)}$ is the frequency of model transition from state i in state j at $k = \overline{1, N}$ steps and $\sum_{(j)} \omega_{ij}^{(k)} = m_{i}^{(k)}$. Thus, the proposed approach makes it possible to assess the degree of adequacy of the developed network models based on temporary Petri nets to consider the prehistory of the ongoing processes and to justify the choice of the route thickness.

5 Model Evaluation and Discussion

An example of the formation of input data for SIA was considered. The process of migration to a modern hardware and software platform was considered. This process was conducted in the accounting system of the employment of the working population of the Kharkiv region. This system was developed in the mid-90s of the 20th century in the COBOL programming language. The performance of the system suited the regional administration until the spring of 2020. However, the system has ceased to cope with the increased load caused by the following reasons due to the pandemic caused by COVID-19.

- The unemployment increase;
- growth of temporarily unemployed population due to lockdown;
- a significant reduction in labor migration to European countries. This condition led to internal migration from neighboring regions to the city of Kharkiv. However, all modern

hardware and software platforms reviewed require a migration from COBOL to a modern programming environment. As noted in the first section, this process is complex and costly. Therefore, applying SIA is necessary. This system is functional. The characteristics of its components are known. Creating a network model of the process of functioning of this system based on temporary Petri nets is possible.

The main software complexes of the employment accounting system are as follows:

ω_1 is the receiving of external requests and internal requests;

ω_2 is the setting up of system services for processing external requests;

ω_3 is the synchronization of required applications;

ω_4 is the checking of the components of requests and requests, as well as their elements;

ω_5 is the decomposition of orders and requests;

ω_6 is the formation of the application pool;

ω_7 is the activation of required applications;

ω_8 is the setting up of system services to process internal requests;

ω_9 is the synchronization of the generated transaction;

ω_{10} is the combining of preliminary results;

ω_{11} is the organization of access to the database;

ω_{12} is the direct work with the database.

Trace data were used to build network models. The models described the process of functioning of the system of accounting for the employment of the able-bodied population. The data were obtained from the results of the system functioning under load in May 2020. The input data of the model are the characteristics of the software systems and the results of the Software trace. Tracing data are used to construct a matrix of transition probabilities. The package model is defined by an event graph. The graph is described by a matrix of transition probabilities. The matrix is based on the results of the analysis of trace data and is shown in Table 2.

Figure 4 shows a temporary Petri net that describes the packet model. The subscript in the transition designation determines the type of request. Dotted lines indicate group transitions. The probability of its triggering is indicated above each transition in the group. The validity of the primary model was checked. The simulation results were compared with the operation of a real outdated system. The characteristics of the network nodes, corresponding to the characteristics of real nodes, were selected. Thus, the frequency characteristics of the transition of the system from the selected state to all other states for K steps were obtained.

The chi-squared test was used $\chi_n^2(k)$ to assess the adequacy of the model, where $k(k \in \overline{1,K})$ is the number of steps. For $K = 100$, the following results were obtained: $\chi_n^2(10) = 1.09$, $\chi_n^2(20) = 1.12$, $\chi_n^2(30) = 2.23$, $\chi_n^2(40) = 3.12$, $\chi_n^2(50) = 5.34$, $\chi_n^2(60) = 7.21$, $\chi_n^2(70) = 10.31$, $\chi_n^2(80) = 16.30$, $\chi_n^2(90) = 23.45$, and $\chi_n^2(100) = 34.37$.

On the basis of these data, the violation of adequacy increases with increasing K. The main reason is the assumption that the transition processes are Markov. Considering the background of the calls is necessary to build a more adequate model in this case.

In the system under consideration, the query execution time was most often distributed according to one of the three laws: deterministic, equable, and exponential. An estimate of the deviation of the parameters of the distribution law of the execution time of a query during modeling from the real one is given in Table 3.

Application of the criterion χ^2 allowed for obtaining a quantitative assessment of the conformity of the behavior of the model to the real system. An increase in the degree of reliability of the simulation results is confirmed by experimental data obtained under various assumptions about the residence time of the system in different states. The experiments conducted indicate the possibility of tuning the optimal parameters of the developed method on the basis of the control sample. In this case, the dependence of the size R_b confidence evaluation interval $\hat{b}$ is the slope of the predictive line from the value and N is the number of measurements must

Table 2 Transition probability values between software systems of the employment accounting system.

	Y											
X	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}	ω_{11}	ω_{12}
ω_1	—	0.34	—	—	—	—	—	0.66	—	—	—	—
ω_2	—	—	1	—	—	—	—	—	—	—	—	—
ω_3	—	—	—	1	—	—	—	—	—	—	—	—
ω_4	0.01	—	0.01	0.56	0.07	0.22	0.08	—	0.03	0.02	—	—
ω_5	—	—	—	1	—	—	—	—	—	—	—	—
ω_6	—	—	—	—	—	—	1	—	—	—	—	—
ω_7	—	—	—	1	—	—	—	—	—	—	—	—
ω_8	—	—	1	—	—	—	—	—	—	—	—	—
ω_9	—	—	—	1	—	—	—	—	—	—	—	—
ω_{10}	—	—	—	—	—	—	—	—	—	—	1	—
ω_{11}	0.27	—	0.32	—	—	—	—	—	—	—	—	0.41
ω_{12}	—	—	—	—	—	—	—	—	—	—	1	—

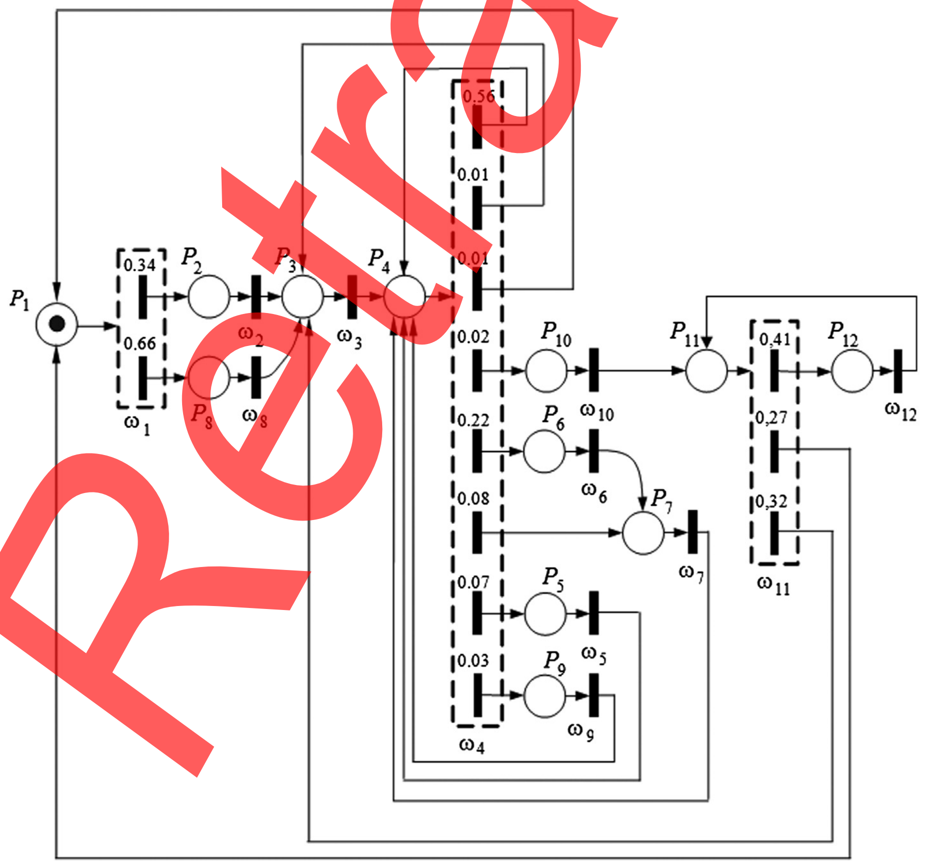


Fig. 4 Graph of the functioning employment accounting systems.

Table 3 Estimation of mathematical expectation and variance on the basis of the results of the simulation for 100 requests.

	Real system		Model	
Distribution law query execution time T	M	D	M	D
Deterministic, $T = 0.01$s	1.74	2.47	1.96	3.45
Equable, $T = [0, 0.01]$	0.96	0.53	1.05	0.99
Exponential, $M(T) = 0.01$	1.88	2.42	2.05	3.36

be considered. The plot of such dependence for various constant values of the variance estimate D_x is shown in Fig. 5.

As shown in the graphs, in most practical time series forecasting cases, a value of N should not exceed 30 to 35. The choice of a larger value to optimize the size of the confidence interval of estimates is not justified, given the main task of the forecast method. The developed network model was used to select the option for the migration accounting system of employment of the working population of the Kharkiv region. Four options were proposed. The options were evaluated in accordance with three criteria: system performance, migration cost, and migration time. The option with the highest performance was selected (variant 3). In this case, a limited cost (Max_2) and time of migration (Max_3) were used. The simulation results are shown in Fig. 6.

A significant limitation of the proposed model is that, with an increase in the number of steps in the iterative process, the degree of model adequacy decreases rapidly.

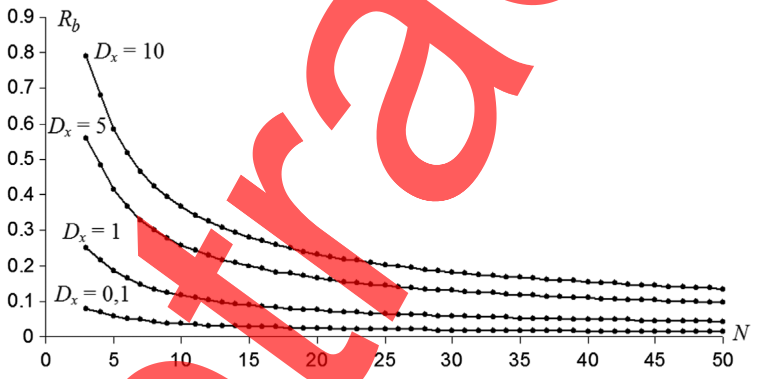


Fig. 5 Dependence of the size of the confidence interval on the number of measurements at $\alpha = 0.05$.

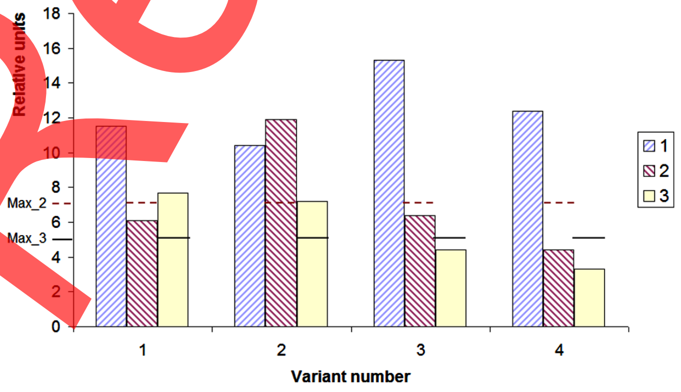


Fig. 6 Results of the analysis of migration options: 1—system performance; 2—migration cost; 3—migration time.

6 Conclusion

A method of the synthesis of network models is proposed. Traces of events in the simulated net are described using deterministic temporal Petri nets. The method is used to develop a network workload model. An approach to assessing the adequacy of the developed model is proposed. An example of evaluating the adequacy of the Markov model of the behavior of a system with discrete states is given. An example of building a model of functioning of a package of composite applications is considered. The package is used for linguistic text processing in the software package. A simulation model of the process is considered to determine the degree of adequacy of the model. With an increase in the number of steps in the iterative process, the degree of model adequacy decreases rapidly. The reason for the violation is the failure to consider the history of the process. For small values of K, the background of the process is ignored. When increasing K, removing the assumption that the process is Markov is necessary. However, this condition significantly complicates the model. Therefore, a direction of further research is to determine the minimum length of the prehistory for a given degree of adequacy. Future research will also include developing a similar model for non-Markovian processes in the future.

References

1. G. Lewis et al., "The service-oriented migration and reuse technique (Smart)," in *13th Int. Workshop Software Technol. and Eng. Pract.*, Budapest, Hungary (2005).
2. T. Erl, *SOA Design Patterns*, p. 864, Pearson Education, USA (2008).
3. S. Newman, *Building Microservices: Designing Fine-Grained in Systems*, p. 280, O'Reilly Media, Inc., USA (2015).
4. R. Khadka et al., "Does software modernization deliver what it aimed for? A post modernization analysis of five software modernization case studies," in *IEEE Int. Conf. Software Maintenance and Evol.*, IEEE, pp. 477–486 (2015).
5. M. Abdellatif et al., "A taxonomy of service identification approaches for legacy software systems modernization," *J. Syst. Softw.* **173**, 110868 (2021).
6. M. Abdellatif et al., "State of the practice in service identification for SOA migration in industry," *Lect. Notes Comput. Sci.* **11236**, 634–650 (2018).
7. M. Mozhaiev, "Modeling nonlinear elements of critical computer network," *Adv. Inf. Syst.* **4**(4), 27–32 (2020).
8. P. Franti, "Efficiency of random swap clustering," *J. Big Data* **5**, 13 (2018).
9. S. Bulba, "Composite application distribution methods," *Adv. Inf. Syst.* **2**(3), 128–131 (2018).
10. V. Mukhin et al., "Decomposition method for synthesizing the computer system architecture," *Adv. Intell. Syst. Comput.* **938**, 289–300 (2020).
11. H. Kuchuk et al., "Adaptive compression method for video information," *Int. J. Adv. Trends Comput. Sci. Eng.* **8**(1.2), 66–69 (2019).
12. I. Ruban, V. Martovytskyi, and N. Lukova-Chuiko, "Approach to classifying the state of a network based on statistical parameters for detecting anomalies in the information structure of a computing system," *Cybern. Syst. Anal.* **54**(2), 302–309 (2018).
13. A. A. Lima et al., "Optimized artificial neural network for biosignals classification using genetic algorithm," *J. Control Autom. Electr. Syst.* **30**(3), 371–379 (2019).
14. V. Tkachov, M. Hunko, and V. Volotka, "Scenarios for implementation of nested virtualization technology in task of improving cloud firewall fault tolerance," in *IEEE Int. Sci.-Practical Conf. Probl. Infocommun., Sci. and Technol.*, Kyiv, Ukraine, pp. 759–763 (2019).
15. G. A. Kuchuk, Y. A. Akimova, and L. A. Klimenko, "Method of optimal allocation of relational tables," *Eng. Simul.* **17**(5), 681–689 (2010).
16. G. Kuchuk et al., "Improving big data centers energy efficiency: traffic based model and method," in *Green IT Engineering: Social, Business and Industrial Applications, Studies in Systems, Decision and Control*, V. Kharchenko, Y. Kondratenko, and J. Kacprzyk, Eds., Vol. **171**, pp. 161–183, Springer, Cham (2019).

17. V. Merlac et al., "Resources distribution method of university e-learning on the hypercovergent platform," in *Conf. Proc. IEEE 9th Int. Conf. Dependable Syst., Service and Technol.*, Kyiv, pp. 136–140 (2018).
18. G. Kuchuk, S. Nechausov, and V. Kharchenko, "Two-stage optimization of resource allocation for hybrid cloud data store," in *Int. Conf. Inf. and Digital Technol.*, Zilina, pp. 266–271 (2015).
19. S. R. Lee, "Dispersion-managed links formed of SMFs and DCFs with irregular dispersion coefficients and span lengths," *J. Inf. Commun. Convergence Eng.* **16**(2), 67–71 (2018).
20. S. Kianpisheh and R. H. Glitho, "Cost-efficient server provisioning for deadline-constrained VNFs chains: a parallel VNF processing approach," in *Proc. 16th IEEE Annu. Consumer Commun. and Networking Conf.* (2019).
21. S. Semenov, O. Sira, and N. Kuchuk, "Development of graphic analytical models for the software security testing algorithm," *Eastern-Eur. J. Enterprise Technol.* **2**(4), 39–46 (2018).
22. D. Jörg and J. Gabriel, "What is a Petri net? Informal answers for the informed reader," *Lect. Notes Comput. Sci.* **2128**, 1–25 (2001).
23. S. Semenov and C. Weilin, "Testing process for penetration into computer systems mathematical model modification," *Adv. Inf. Syst.* **4**(3), 133–138 (2020).
24. P. Marius, M. Cuc, and I. Oncioiu, "Analytical modelling of share price value using computational intelligence methods," *Int. J. Comput. Commun. Control* **15**(4), 1–12 (2020).
25. W. M. P. van der Aalst, C. Stahl, and M. Westergaard, "Strategies for modeling complex processes using colored Petri nets," *Lect. Notes Comput. Sci.* **7**, 6–55 (2013).
26. H. Khudov et al., "Optimization of joint search and detection of objects in technical surveillance systems," *Adv. Inf. Syst.* **4**(2), 156–162 (2020).
27. R. Valk, "Object Petri nets," *Lect. Notes Comput. Sci.* **3098**, 819–848 (2004).
28. Q. Ye and W. Zhuang, "Distributed and adaptive medium access control for internet-of-things-enabled mobile networks," *IEEE Internet Things J.* **4**(2), 446–460 (2017).
29. M. F. Geronimo et al., "A multiagent systems with Petri Net approach for simulation of urban traffic networks," *Comput. Environ. Urban Syst.* **89**, 101662 (2021).
30. M. Mozhaiev and P. Buslov, "Method of modeling of a social profile using big data structure transformation optimization," *Adv. Inf. Syst.* **5**(1), 12–17 (2021).
31. N. Bacanin et al., "Task scheduling in cloud computing environment by grey wolf optimizer," in *Telecommun. Forum*, IEEE, Belgrade, Serbia (2019).
32. T. Bezdan et al., "Enhanced flower pollination algorithm for task scheduling in cloud computing environment," *Lect. Notes Networks Syst.* **141**, 163–171 (2021).
33. T. Bezdan et al., "Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm," in *Intelligent and Fuzzy Techniques: Smart and Innovative Solutions*,C. Kahraman et al., Eds., pp. 718–725, Springer, Cham (2021).
34. J. Li et al., "Modeling and analysis of network control system based on hierarchical coloured Petri net and Markov chain," *Discrete Dyn. Nat. Soc.* **2021**, 9948855 (2021).

Amin Salih Mohammed is an assistant professor at Lebanese French University. He has more than 15 years of teaching and research experience. He is an active researcher and acted as a resource person for various workshops and faculty development programs organized by different institutions. He completed his undergraduate and postgraduate engineering degrees and his PhD in computer engineering from Kharkiv National University of Radio Electronics, Kharkiv, Ukraine. His research fields include computer networks, wireless networks, and cloud computing. He has published more than 40 research articles in various reputed international journals indexed in SCI & Scopus services and reputed conferences. He is an active member of IEEE.

Andriy Kovalenko is an associate professor at Kharkiv National University of Radio Electronics. He received his BE (CSE), MTech (IT), and PhD (CSE). He received his Dr.Sc. degree in computer science and engineering in the field of computer systems and components at the National Technical University "Kharkiv Polytechnical Institute." He has over 18 years of experience in training, consultancy, teaching, and placements. His current areas of research cover computer systems and networks, industrial instrumentation, and control systems, as well as their cyber security and dependability. He has published more than 190 research papers (including several monographs). He has guided many research projects, and he is the coeditor of a scientific journal and a program committee member of several international conferences.

Nina Kuchuk graduated from the National Technical University "Kharkiv Polytechnic Institute" with a degree in economic cybernetics in 2011. She defended her thesis and received her PhD in 2014. Since 2014, she has been working as an associate professor at V. Karazin Kharkiv National University. Since 2019, she has been working as an associate professor in the Department of Computer Engineering and Programming of National Technical University "Kharkiv Polytechnic Institute." Her research interests include information technology and radio communication.