A Method for Estimating the Execution Time of Computer Programs in Automated Information Transmission Systems

Mirzaeva Malika, Suleymanov Anvar, and Mamatov Narzullo

***Abstract*—This article presents the development of mathematical models for evaluating the execution time of computer programs within two distinct classes of computational resources (CR) in Automated Communication Systems (ACS): (i) systems characterized by a multistage program execution process, and (ii) systems operating with finite load source devices and unreliable components. The primary objective is to derive analytical expressions for computing key time-related characteristics—such as execution delay, waiting time, and system throughput—by modeling these CRs as multi-phase queueing systems. Two time allocation strategies are considered: fixed-priority scheduling and cyclic weighted execution. The models capture both the structural complexity and timing behavior under varying processor loads and service priorities. Comparative simulation results confirm the effectiveness and accuracy of the proposed models, highlighting their advantages over conventional static estimation approaches. The proposed framework provides a robust foundation for optimizing scheduling policies and improving timing predictability in real-time control environments, particularly in critical automated communication networks and embedded systems.**

***Index Terms*—mathematical models, computational resources, time characteristics, automated information transmission systems, and processors.**

I. Introduction

SIGNIFICANT attention is paid to evaluating and enhancing the efficiency of communication networks and systems worldwide. Improving their efficiency is one of the key factors contributing to advancing a country's socio-economic development program. Modern control systems are typically characterized by significant complexity, many elements, and high intensity of the information processes occurring within them. The organization of control systems is impossible without comprehensive automation based on computational technology tools that enable the implementation of complex algorithms requiring the execution of numerous actions within a limited time frame and the operational analysis of the current state of the control object. Practice shows that modern computing machines address management tasks that comprise 80 percent logical control tasks, 10–15 percent regulation tasks, and 5–10 percent management tasks utilizing complex specific algorithms.

The features of the functioning process of computing resources in Automated Information Transmission Systems (AITS), as well as the limitations of processor performance, the limited number of peripheral nodes, the stochastic nature of program execution requests, and, consequently, the formation of request queues at various stages of the execution of algorithms, necessitate the development of models for computing resources as queuing systems that accurately reflect the functioning process of AITS. Constructing these models as queuing systems allows the assessment of key quality characteristics of service and the performance of computing resources, considering specific disciplines and service regimes. On this basis, it is possible to calculate the main time characteristics of computing resources given the volume and intensity of functional tasks assigned to the computing resources of AITS.

Among the types of computing resources in an Automated Information Transmission System (AITS) and their operational modes discussed in article [1], two types of computing resources can be distinguished based on the nature of their operational processes and the associated organization of program execution processes. In particular, two major categories of computing resources in AITS are identified based on program execution behavior:

- Type I: Multistage program execution with strict timing and dispatching constraints.
- Type II: Processing requests from finite load sources, where rapid response is crucial but startup delay is tolerable.

Efficient configuration and routing strategies significantly impact the performance of Automated Information Transmission Systems (AITS). For instance, multi-objective optimization frameworks have been explored in network design to enhance routing efficiency, as seen in [2]. While such methods primarily focus on data transmission, our work shifts attention to program-level execution within computing nodes.

Currently, no well-established solutions account for these unique operational modes. Existing estimation methods often ignore multi-phase program execution and the stochastic and fault-prone behavior in Type II systems. This motivates the development of more accurate models for execution time estimation and performance analysis in such environments.

Manuscript received September 23, 2025; revised January 15, 2026. Date of publication July 8, 2026. Date of current version July 8, 2026. The associate editor prof. Sven Gotovac has been coordinating the review of this manuscript and approved it for publication.

Authors are with the Tashkent Institute of Irrigation and Agricultural Mechanization Engineers National Research University, Tashkent, Uzbekistan (e-mails: malikamirzaeva01@gmail.com, saahumans@gmail.com, m_narzullo@mail.ru).

Digital Object Identifier (DOI): 10.24138/jcomss-2025-0093

The main contributions of this paper are summarized as follows:

- We identify two categories of control computing resources (Type I and Type II) in Automated Information Transmission Systems (AITS) with distinct operational requirements.
- We develop mathematical models for both categories, incorporating queueing theory with fixed and cyclic priority-based time allocation schemes.
- We derive analytical expressions for computing the average waiting time, execution time, and service delay under varying processor loads and priority levels.
- We perform simulation experiments to validate our models and visualize performance trade-offs under multiple load and priority conditions.
- This study establishes a link between theoretical queuing models and practical evaluation in AITS environments.

The remainder of the paper is organized as follows: Section II reviews the related works. Section III introduces the modeling methods and mathematical formulations. Section IV presents simulation results and discussion. Section V concludes the paper and outlines future work directions.

II. Related Works

Execution time estimation in real-time, embedded, and distributed systems has been intensively studied over the last few decades, with several mainstream directions emerging [10], [24]-[26], [30], [31], [35], [36].

A first group of methods focuses on worst-case execution time (WCET) and schedulability analysis in real-time systems, typically relying on static code and architectural analysis. Kozyrev reviews the execution time estimation challenges and summarizes several solution approaches for real-time control computing systems, emphasizing upper-bound guarantees for safety-critical applications [35]. Classical and modern performance modeling texts develop queueing-based abstractions of processing resources and link them to scheduling analysis in computer and communication systems [3], [10]. However, these WCET-oriented and analytical methods usually treat the computing node as a single-stage resource and do not explicitly capture multi-stage program flows or finite-source behavior of control tasks.

A second line of work leverages measurement-based and machine learning–based prediction. Porjaroenphan et al. argue that accurately predicting the execution time of a task before execution is unrealistic in highly dynamic environments; instead, they estimate execution times from historical runs using feature vectors and K-Nearest Neighbors regression [36], [37]. Similar ideas have been applied to jobs and workflows in large-scale and high-performance computing, where machine-learning models are trained on past executions to predict job completion times or queue waiting times [8], [31], [24]. These approaches adapt well to run-time variability and heterogeneous infrastructures, but they generally lack closed-form analytical expressions and are not tailored to the specific operational modes of Automated Information Transmission Systems (AITS) or to strict schedulability guarantees.

Queueing-theoretic approaches form a third important direction. Standard queueing models for M/M/1 and related systems, and their extensions to finite-capacity, multi-server, and multi-priority queues, provide analytical formulas for waiting times, loss probabilities, and server utilization [3], [10], [11], [25], [26], [29], [30], [33], [34]. These models have been generalized to multi-queue and multi-class settings, including unreliable and retrial queues, working vacations, and adaptive priority scheduling, and have been applied to communication and computing networks [25], [26], [28], [29], [30], [32], [33], [34]. Such work typically assumes generic servers and stationary or stylized traffic models, rather than control computing resources operating under specific AITS modes, finite sources, and structured multi-phase control programs.

These methods are discussed solely to clarify the scope and limitations of the proposed approach. Modern operating systems and resource managers incorporate fixed-priority, round-robin–type, and more advanced priority-based scheduling, and have been applied in automated control, edge, and telecommunication systems, including multilevel and feedback-based task management [10], [24], [27]-[29]. However, few works integrate both the structural complexity of multi-phase control computing resources and their timing behavior under varying processor loads and mixed time-allocation disciplines. The present study addresses this gap by formulating a unified queueing-based execution time model for controlling computing resources in AITS, explicitly capturing multi-stage control flows, finite-source effects, and two alternative time allocation schemes, and by analyzing its performance using both analytical and simulation techniques. In contrast to classical WCET and schedulability methods, which typically treat the computing node as a single-stage resource, the proposed framework explicitly models multi-phase control flows and finite-source CCR behavior in AITS. At the same time, unlike measurement- or machine-learning-based predictors that lack closed-form guarantees, the queueing-theoretic models provide analytical expressions for waiting time, execution delay, and busy period under both fixed-priority and cyclic weighted scheduling. This combination of AITS-specific structure, priority-based time allocation, and closed-form timing analysis distinguishes the present work from existing execution time estimation approaches.

Kozyrev provides a review of execution time estimation challenges and summarizes several known solution approaches [35]. This work focuses on estimation in AITS with the aim of defining an upper bound for the computation execution time that cannot be exceeded. In contrast, our work attempts to estimate execution time in AITS under typical operating conditions.

Phinjaroenphan et al. [36] note that it is unrealistic to assume that the execution time of a task assigned to a node can be accurately predicted before actual execution. In their method, it is not necessary to know the internal design or algorithm of the application. The estimation is based on historical observations of task executions. For each execution, they collect a vector of predictors, each of which impacts execution time. The estimation method they use is the K-Nearest Neighbors algorithm [37].

III. Methods

A. Classification of Control Computing Resources

This article presents methods for calculating the main time characteristics of control computing resources in automated information transmission systems (AITSs), developed based on an analysis of their specific operational features.

For control computing resources (CCR) of the first type, a methodology is proposed to study their functioning process, based on representing CCR as a multistage open queueing system with a queue before each stage and priorities assigned to individual stages.

For control computing resources (CCR) of the second type, their functioning is analyzed as single-phase closed multi-channel queueing systems with unreliable components. Relationships are derived that allow the calculation of the performance characteristics of such CCR under various operating conditions.

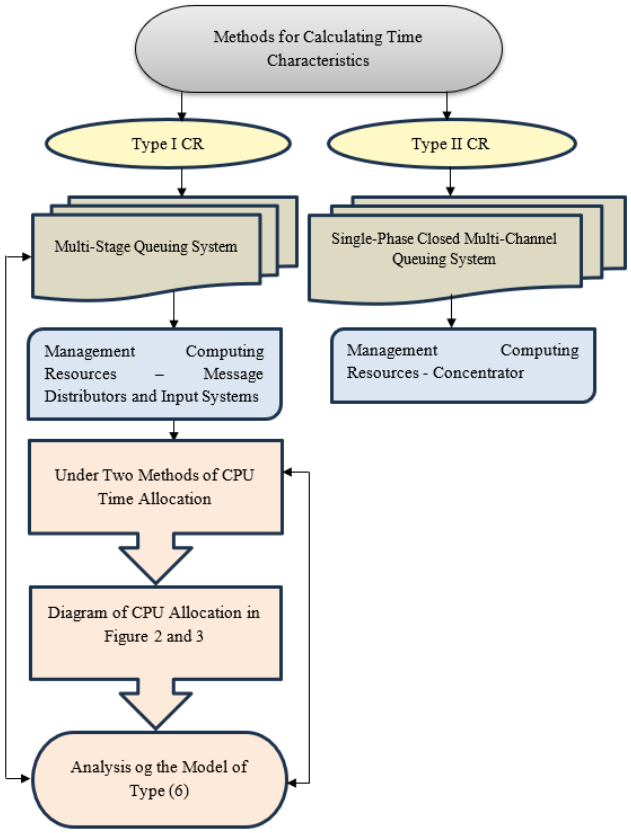

Fig. 1. An Overview of the Research Process in Modeling and Analyzing Control Computing Resources

B. Time Allocation Methods for CCR of Type I

To develop mathematical models describing the functioning of control computing resources (CCR) of the first type, primarily including message distribution CCR and input CCR, we examine methods of allocating machine time within the computing resources of an Automated Information Transmission System (AITS) during multistage program execution.

In the first method of machine time allocation Fig 2, subroutines are assigned numbers based on the increasing order of allowed delays for their initiation. A separate queue is organized for each subroutine request $Q_i(i = \overline{1,K})$. When there are requests for the execution of several subroutines, the subroutine with the smallest number is selected for execution. The first method is a type of fixed priority scheduling [4]–[7]. The priority of the subroutines is determined by the values of $t_i max$. In this regard, the priority $i-$ subroutines with higher priority $j-$ subroutines, $t_i max < t_j max(i, j = \overline{1,K})$.

This method is relatively simple to implement; however, it does not take into account the relationships between the permissible delays for the initiation of different subroutines [8], [9]. Moreover, implementing this method incurs considerable processor time due to the overhead involved in selecting which subroutine to execute.

In the second method, these shortcomings are addressed by establishing a specific cyclic order to check the availability of requests for different subroutines. When a request to execute a particular subroutine is detected among several, that subroutine is executed the required number of times. Only after this is the system allowed to check for other requests [1]–[3]. In this method (Fig. 2 b), subroutines are activated at times determined by the processor cycle, and their priorities are defined by specifying the number of requests ω_i assigned to each subroutine $(i = \overline{1,K})$ within one cycle.

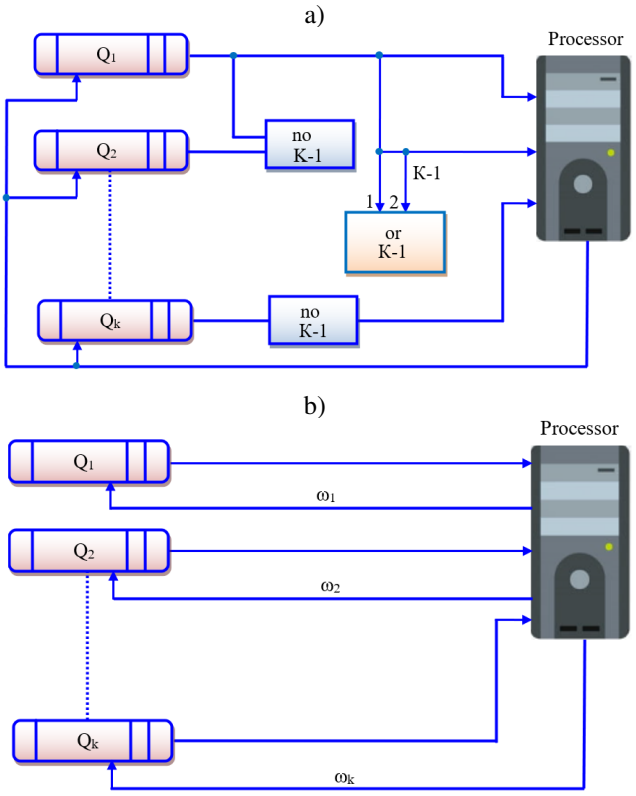

Fig. 2. Subroutine Execution Algorithm in Processor Operation Mode: (a) Fixed Priority, (b) Cyclic Weighted Scheduling

This method takes into account not only $t_i max < t_j max$ for the subroutines i and j $(i, j\overline{1,K})$, but also the ratio $t_j max/t_i max$. Regardless of the presence of requests for the execution of multiple subroutines, the one for which the processor reaches the moment of the request is activated

[10], [11]. Thus, considering the ratio t_jmax/t_imax in this discipline allows the priority of the $i-$ subroutine t_jmax/t_imax times higher than the priority of the $i-$ subroutine, where t_imax and t_jmax are the delay times for the respective subroutines. Suppose the processor's work cycle in the computing system is described by the sequence: 1,2,1,3,1,2,1,4,1,2,1,3,1,2,1,5,1,2,1,3,1,2,1,4,1,2,1,1,3,1,1,2,1. Where $K = 5$, and the number of requests per subroutine within one processor cycle be defined as: $\omega_1 = 16$, $\omega_2 = 8$, $\omega_3 = 4$, $\omega_4 = 2$, and $\omega_5 = 1$. Then, the total number of requests within one complete cycle (cycle length) is given by:

$$N = \sum_{i=1}^{K} n_i = 31.$$

C. Modeling of Multi-Phase Queueing System

Based on the features of Type I control computing resources (CCR) in an Automated Information Transmission System (AITS), they are modeled as a general K-phase open service system. Each phase i $(i = \overline{1,K})$ has its own queue and service device C_i, and relative priorities are assigned among phases.

The flow of incoming requests corresponds to demands for executing various programs. The service performed in phase i represents the execution of the i-th subroutine $(i = \overline{1,K})$.

The multiphase structure of this service model reflects the fact that executing any complete program involves passing through a specific sequence of subroutines [12]. After each subroutine is completed, the system must again determine which subroutine — and thereby which program — to execute next.

For each request to execute the $j-$ program $(j = \overline{1,R})$, the sequence of service phases is predefined. We refer to this sequence as the service path of the request. Consequently, the multi-phasic servicing systems under consideration are systems with R paths for servicing requests. In the proposed model, the execution of the $j-$ program in computing resources corresponds to the request passing through the K_i service phases.

If the $i-$ phase belongs to the m_i service paths, the request flow intensity in this phase is determined by summing the corresponding intensities. If λ^2- is the intensity of the request flow for the $j-$program [13], [14]. Then, if the $i-$ phase $(i = \overline{1,K})$ is traversed by requests from m_i service paths $(m_i \leq R)$, the request flow intensity at this phase is determined by adding all paths that pass through it, i.e.,

$$\lambda_i = \sum_{j \in m_i} \lambda^j. \tag{1}$$

Among the proposed models, we distinguish between single-node and multi-node service systems. A system with the same service device at all phases K is called a single-node system. If requests are serviced by different devices (or groups of devices) at different phases, the system is referred to as a multi-node system. The flow of incoming requests for the execution of the programs can be heterogeneous, with N priority levels for different requests. Additionally, individual programs may have different priorities compared to other programs. In both cases, priorities are relative, meaning they only apply when selecting subroutines from the queue before the ith phase [15]–[17]. However, we will focus on models with a homogeneous incoming request flow.

The condition for the stationary of the functioning process of type I controlling computing resources (CCR) as a single node $K-$ phase service system with R service paths for requests is as follows:

$$\rho = \sum_{i=1}^{K} \lambda_i t_i = \sum_{i=1}^{K} t_i \sum_{j \in m_i} \lambda^j < 1 \tag{2}$$

where ρ is the service load coefficient for the device in all phases; and t_i is the service time in the ith phase.

For a multi-node $K-$phase system with R service paths, the stationary condition can be written as:

$$\rho_i = t_i \sum_{j \in m_i} \lambda^j / C_i < 1 \tag{3}$$

where ρ_i is the load factor of the ith phase?

The necessity to account for the specific features of the structure and organization of the functioning process in the computing resources of an Automated Information Transmission System (AITS) of various types dictates the presence of several specific models of computing resources that fall into this class of models [18], [19]. We will highlight those that correspond to the modes of control computing resources (CCR) that are practically relevant.

The model of control computing resources (CCR) with a single level of information processing (single-level control computing resources) in the modes of Operational Readiness (OR) and Diagnostic (D) for CCR in general, and Resource Scheduling and Resource Control for the CCR subsystem, consists of single node Kth phase systems with R service paths and a service discipline with fixed relative phase priorities [1]. This model is illustrated in Figure 3.

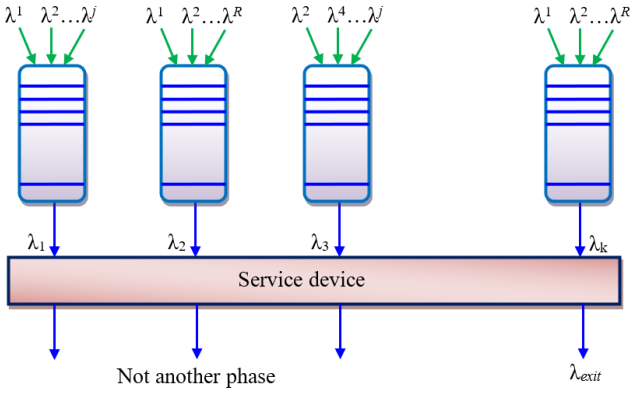

Fig. 3. Diagram of the Model of Control Computing Resources (CCR) with Single-Level Information Processing

The model of control computing resources (CCR) with two or more levels of information processing (multilevel control computing resources) in the Subsystem Operation mode is represented by single-line Kth phase systems with R service

paths and the same service disciplines. This model is illustrated in Fig. 4.

Thus, the object of study is a single-node or multi-node open multiphase system with waiting, phase priorities for service, and multiple service paths.

To study such service systems, in addition to the service discipline, it is necessary to characterize:

- the distribution function of the inter-arriving times of requests to the ith phase $A_i(t)$;
- the distribution function of service times for requests at each phase of the system $B_i(t)$;
- the quality of service for requests.

Regarding incoming streams, the following propositions can be made. Any of the R independent incoming streams may represent a regular stream [20], [21]. In the general case, the duration of the time intervals between the moments of request arrivals for program execution are random variables with certain distribution laws. For most phases, the following assumptions can be made:

$$A_i(t) = 1 - e^{-\lambda_i(t)}.$$

The assumption of a Poisson nature for incoming streams is justified by the fact that a Poisson process, being the limiting case in the class of Erlang processes, creates the most challenging conditions for the servicing system. At the same time, for a small number of CCR models, incoming streams at some phases of the system can be significantly different from the simplest case [22]. These include non-standard streams, streams that are a sum of regular and Poisson processes, and streams resulting from the superposition of several individual processes with various types of request arrival laws.

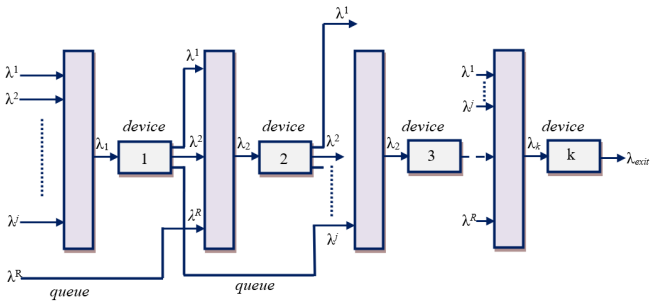

Fig. 4. Diagram of the Model of Control Computing Resources (CCR) with Single-Level Information Processing

By the service time of requests at the ith phase, we mean the time interval from when the $i-$th subroutine starts to when its execution is completed. For some phases, it can be assumed that the distribution function of the service time of requests at the ith phase has the following form:

where μ_i is the average service time of requests in the i th phase ($\mu_i = 1/overlinet_i$). However, in the general case, the service time of requests in each phase is characterized by a degenerate distribution. The programs of the control computing resources for the Automated Information Transmission System (AITS) can be fully characterized by a constant execution time [25]. Therefore, the execution time of the $i-$th subroutine $(i = \overline{1, K})$ can be considered a constant value.

The main characteristics of the quality of service for requests can be considered as follows:

- the distribution function of the time a request spends in the queue before the $i-$th service phase, $C_i(t)$, $(i = \overline{1, K})$;
- the distribution function of the time a request spends in the $i-$th service phase $D_i(t)$;
- the distribution function of the time a request spends on the $j-$th service path, $E_j(t)$, $(j = \overline{1, R})$.

However, for evaluating the temporal characteristics of the functioning of control computing resources, it is sufficient to consider the expected values of these random variables [23]. If the average time a request spends in the queue before the ith service phase is denoted by $\overline{\tau_{oi}}$, then the average time a request spends at the ith service phase is

$$\overline{\tau_i} = \overline{\tau_{oi}} + \tau_i, \tag{4}$$

the average time a request spends on the jth service path

$$\overline{\tau_j} = \sum_{j \in m_i} \overline{\tau_j}. \tag{5}$$

These characteristics allow for determining the main temporal characteristics of the control computing resources of the Automated Information Transmission System (AITS). For example, the average time a request spends in the queue before the $i-$th phase $\overline{\tau_{oi}}$ allows for the evaluation of the delay time before the execution of individual subroutines begins. The average time a request spends on the $j-$th service path allows for the evaluation of the program execution time in the control computing resources of the Automated Information Transmission System (AITS).

Partially using the notation of D. Kendall [24], the introduced class of queueing models can be denoted as follows:

$$K|R|\overrightarrow{GI}|\overrightarrow{D}|C_i \geq 1|0 < r_i \leq \infty|f_{s.ph}.$$

This means that the service system consisting of K service phases with R service paths receives a request flow with an arbitrary distribution function for the arrival times of requests (GI), with different arrival intensities $\lambda_1\lambda_2...\lambda_K(\overrightarrow{GI})$ for each phase, a degenerate distribution for the service times in each phase $(\overrightarrow{D})$, the number of waiting places at each phase is unlimited $(0 < r_i \leq \infty)$, and the number of servers at the $i-$th phase is $C_i(i = \overline{1, K})$. For multi-node systems $C_i > 1$, the model is for single-node systems $C_i = C = 1$. The service discipline for requests follows phase-based priorities $f_{s.ph}$.

Multiphase systems with waiting have been studied by R. Jackson and J. Jackson [25]. In these studies, assumptions included a Poisson incoming flow, an exponential distribution of service times, and an arbitrary order of transition for requests from the $i-$th phase to the $(i+1)-$th phase (requests move from one phase to the next as servers become available at the next phase).

In the works devoted to the analysis of computing systems as queueing systems [26], computing systems are examined either as multiphase systems with a homogeneous incoming flow, an exponential distribution of service times in each phase, and requests moving to the next phase when a server in

that phase is free, or as single phase systems with Poisson incoming flows N, different service time distributions, and different service disciplines for incoming flows.

The models of control computing resources considered are generally characterized by the following:

1) A multi-phase nature of request servicing;
2) The presence of multiple incoming streams, each managed in each phase by a system with fixed relative priorities of individual request streams;
3) The initial service time for requests for each stream in each phase;
4) A discipline of service with relative priorities of individual phases.

Well-known queueing models have been studied under the assumption of the first condition (multiphase nature) or the second and third conditions (single-phase systems with N incoming streams). It should be noted that a characteristic feature of the considered models is that requests can transition from one phase to another only upon receiving an authorization signal, which is issued according to the adopted service discipline, specifically the phase-based priority system.

Models of the functioning of control computing resources for the Automated Information Transmission System (AITS) with a multistage program execution process are, as shown above, single-node $K-$phase systems with R service paths and phase priorities specified by the two methods discussed earlier, of the type:

$$K|R|\overrightarrow{GI}|\overrightarrow{D}|0 < r_i \leq \infty|f_{s.ph}. \tag{6}$$

Single-line multi-node $K-$ phase systems with R service paths and phase priorities of the type

$$K|R|\overrightarrow{GI}|\overrightarrow{D}|0 < r_i \leq \infty|C_i = 1|f_{s.ph}. \tag{7}$$

Multi-node, multi-line $K-$ phase systems with R service paths and phase priorities of the type

$$K|R|\overrightarrow{GI}|\overrightarrow{D}|0 < r_i \leq \infty|C_i > 1|f_{s.ph}. \tag{8}$$

Key Steps of the Proposed Method

- Model the control computing resources as multistage queueing systems.
- Define two scheduling methods: fixed priority and cyclic weighted execution.
- Derive analytical expressions for delay, queue size, and execution time.
- Validate the timing behavior under different processor loads.

The proposed mathematical models adequately reflect the main features of the operation of control computing resources for the Automated Information Transmission System (AITS) with a multistage program execution process. However, due to their complexity, they are extremely difficult to analyze [27]. To determine the quality of service characteristics for the studied models, we use an approach based on equivalently replacing the considered models with systems for which obtaining characteristics is less challenging.

Consider systems of type (6). Under both methods of time allocation discussed above, this model reduces to a $K-$phase system with a single server (processor) in all K phases, which at any given time can only be engaged in the service of one request in one of the phases.

The transfer of control from one phase to another, governed by a service discipline with relative phase priorities, is not instantaneous but occurs with some delay. This delay arises because the processor requires some time t_i^* to locate the $i-$th phase where there is at least one request $(i = \overline{1,K})$. During this time t_i^*, the processor does not perform any phases. Thus, the impact of scheduling on the quality of the requested service will be reflected in the increase in the average time delay for the start of service in the $i-$th phase $(i = \overline{1,K})$.

To obtain an upper bound for the waiting time before the start of service in the i th phase $t_{oi}(i = \overline{1,K})$, we assume that the request flows in all service phases are Poisson. This approach is justified because, with constant service times, a Poisson flow of requests, after passing through the service system, forms an Erlang flow. The Poisson flow, being the limit case of Erlang flows [28], creates the most challenging conditions for the service system.

We analyze the model of type (6) under the first method of time allocation. Assuming that the request flows at all service phases are Poisson, the $K-$phase queueing system with relative phase priorities becomes equivalent to a single-phase system with K incoming flows, each having relative priorities in service. If the request flow numbers increase with decreasing priority, and the service times for all request flows are constant $t_i(i = \overline{1,K})$, the average waiting time for requests of the $i-$th flow in such a single-phase system can be determined using the known formula [9]:

$$\bar{t}_{oi} = \frac{\sum\limits_{j=1}^{K} \lambda_j t_j^2}{2\left[1 - \sum\limits_{j=1}^{i-1} \lambda_j t_j\right]\left[1 - \sum\limits_{j=1}^{i} \lambda_j t_j\right]}, \tag{9}$$

where λ_j is the intensity of the jth flow.

D. Analytical Derivation of Waiting-Time Expressions

In formula (9), it was assumed that the flow numbers increase with decreasing priority and that the sampling times for requests (dispatching) are zero. In a more general case, it can be considered that the time spent by the processor on sampling any subprogram depends only on its number [29]. The time costs for dispatch t_i^* are thus equivalent to an increase in the duration of the execution of the subprograms t_i. Therefore, using formula (9), we can derive the following expression for the average waiting time before the start of service for a request in the $i-$th phase:

$$\bar{t}_{oi} = \frac{\sum\limits_{j=1}^{K} \lambda_j (t_j + \bar{t}_j^*)^2}{2\left[1 - \sum\limits_{j=1}^{i-1} \lambda_j (t_j + \bar{t}_j^*)\right]\left[1 - \sum\limits_{j=1}^{i} \lambda_j (t_j + \bar{t}_j^*)\right]}, \tag{10}$$

where λ_j is the intensity of the request flow for the execution of the request ith subprogram, t_j is the duration of its execution and $\bar{t}_j^*$ is the average time spent sampling the jth subprogram.

E. Numerical Evaluation Setup

To validate the analytical expressions derived in the previous sections, simulation experiments were conducted. The results were visualized to illustrate the relationship between system parameters (such as device load and number of priority levels) and key performance metrics like waiting time, dispatch delay, and throughput. Figures 4 and 5 illustrate the dependencies of the normalized values of the average waiting time before the start of service for the first method of time allocation on the overall load of the service device (processor)

$$\rho = \sum_{i=1}^{K} \lambda_i (t_i + \bar{t}_i^*).$$

The graphs are constructed for $K = 5$ in $t_i + \bar{t}_i^* = t$ and $\lambda_i = \lambda$. The time spent sampling the ith subprogram was estimated by creating several variants of the subprogram time allocation.

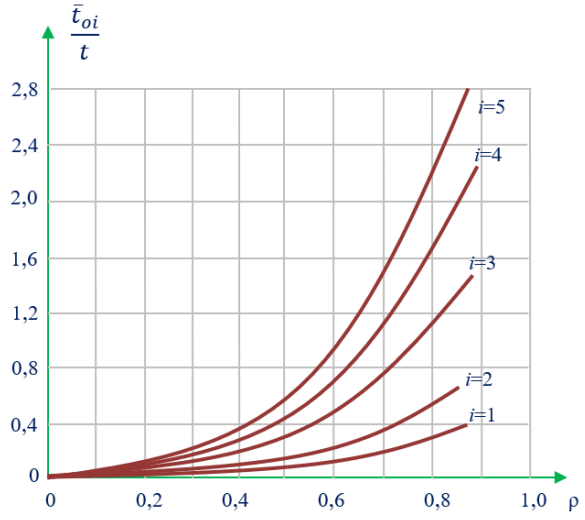

Fig. 5. Graph of the Dependence of the Average Dispatching Time on the Service Device Load(a)

As shown in Figures 5 and 6, the average dispatching time increases with processor load. However, under lower load conditions, differences between priority levels are marginal. As load approaches saturation, the system increasingly prioritizes higher-priority tasks, leading to rapid growth in dispatch time for lower-priority subroutines.

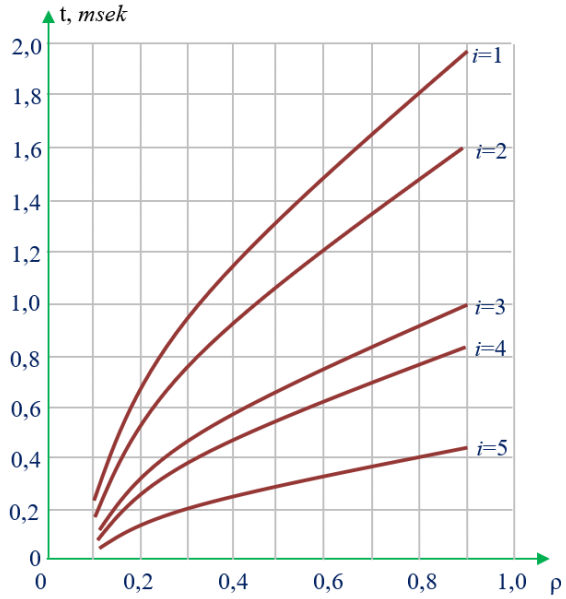

Fig. 6. Graph of the Dependence of the Average Dispatching Time on the Service Device Load (b)

The graphs show that at low levels of overall load $\rho < 1$, the waiting times in the queues before phases of different priorities differ only slightly. At high levels of total load $\rho < 1$, there is a redistribution of waiting time in favor of higher priorities.

In general, the average time spent sampling the ith subprogram is given by

$$\bar{t}_i^* = \bar{t}\,(m_1 + m_2 \cdot i \cdot \bar{r}_i)\,, \tag{11}$$

where $\bar{t}$ is the average execution time of the command in the processor, m_1 is the number of commands for processing the request signal for the presence of requests in queues for one or more phases with a priority not lower than $(i-1)$th, i is the phase (priority class) ordinal number, m_2 is the number of commands for checking the presence of requests in queues for phases of any priority, $\bar{r}_i$ is the average number of requests that may arrive at phases with a priority not lower than $(i-1)$th during the service time of a request at the ith phase.

By $P_1(t_2)$ one denotes the probability that during the service time of a request in the second phase, one or more requests will arrive at the queue before the first phase [30]. This probability is given by

$$P_1(t_2) = 1 - P_0(t_2), \tag{12}$$

where $P_0(t_2)$ is the probability that there are no requests in the queue before the first phase during the time t_2. In the considered model, the incoming flows are assumed to be simple, so

$$P_0(t_2) = e^{-\lambda_1 t_2}. \tag{13}$$

The probability of at least one request arriving in the queue before the first phase during the service time of requests in the second phase is $P_{12} = 1 \cdot P_1(t_2)$. During the service time for the requests in the second and third phases, the number of requests that arrive in the queue before the first phase is $2[1 - e^{-\lambda_1(t_2+t_3)}]$. During the service time for requests in the

second, third,..., ith phases, the number of requests arriving at the queue before the ith phase is $(i-1)[1-e^{-\lambda_1(t_2+\ldots+t_i)}]$. The overall average number of requests in the queue before the first phase is

$$\bar{r}_1 = \sum_{i=2}^{K}(i-1)(1-e^{-\lambda_1 \sum_{n=2}^{i} t_n}). \quad (14)$$

Similarly, to expression (14), expressions can be written to determine the average number of requests in the queue before the second phase

$$\bar{r}_2 = \sum_{i=3}^{K}(i-2)(1-e^{-\lambda_2 \sum_{n=3}^{i} t_n}). \quad (15)$$

The average number of requests in the queue before the ith phase is

$$\bar{r}_i = \sum_{l=i+1}^{K}(l-i)(1-e^{-\lambda_i \sum_{f=i+1} t_f}). \quad (16)$$

Substituting expression (16) into (11), the final expression for the average dispatch time is

$$\bar{t}_i^* = m_1\bar{t}\left[1+\alpha K\sum_{l=i+1}^{K}(l-i)(1-e^{-\lambda_i \sum_{z=i+1}^{l} t_z})\right], \quad (17)$$

where α denotes the ratio m_2/m_1.

For the expression obtained (17), the graphs of the dependence of the average dispatch time on the load of the servicing unit are shown in Fig 6.

Calculations are performed in $K=5$, $a=0.1$, $m_1=10$, $\bar{t}=2$ microseconds.

Analysis of the graphs shows that as CPU load increases, the dispatch time for phases of any priority level increases, with the dispatch time for higher priority phases increasing significantly more compared to the dispatch time for lower priority phases [31]. Similarly, the average queue length before phase i and the mean time spent by requests in phase i and along service path j are computed via (4)–(5).

IV. Results and Discussion

This section presents numerical and simulation results based on the analytical expressions derived in Section III. The impact of processor load, number of phases, and priority configuration on key timing metrics is illustrated and interpreted. Expression (10) is derived for the case where $R=1$. If m_i threads pass through the ith phase $(i=\overline{1,K})$, the average queueing time for requests before the ith phase in a model with fixed relative priorities for the phases, determined by their numbers, can be given by the following formula:

$$\bar{t}_{oi} = \frac{\sum_{i=1}^{K}(t_i+\bar{t}_i^*)^2 \sum_{j\in m_i}\lambda_j}{2[1-\sum_{v=1}^{i-1}(t_v+\bar{t}_v^*)\sum_{j\in m_i}\lambda_j][1-\sum_{v=1}^{i}(t_v+\bar{t}_v^*)\sum_{j\in m_i}\lambda_j]}. \quad (18)$$

The average length of the queue before the ith phase

$$\bar{L}_{oi} = \bar{t}_{oi}\sum_{j\in m_i}\lambda_j. \quad (19)$$

The average time that requests spend in the ith service phase is given by formula (4), while the average time that requests spend in the jth service path $(j=\overline{1,R})$ is determined by formula (5).

We analyze a system's performance characteristics, as described in equation (6), using the second method of machine time allocation.

As discussed previously, the service device can be in one of K states, with state $(i=\overline{1,K})$ corresponding to the ith service phase. After completion of the ith phase, the device transitions to another state according to the service discipline in use. If t_{ii} denotes the time it takes for the device to return to the same i-state from the state $(i=\overline{1,K})$, then the average time for the service device to transition from the i-state back to the same state is given by the following:

$$t_{ii} = \frac{\overline{T}_{cycle}}{n_i}. \quad (20)$$

Here, $\overline{T}_{cycle}$- denotes the average cycle duration of the processor. The probability that the service device is idle, that is, it is not performing any phases, is represented by the following expression

$$1-\sum_{i=1}^{K}\lambda_i t_i = 1-\rho. \quad (21)$$

Expression (21) is valid for any service discipline [23]. During a sufficiently large time interval T, the total time that the service device spends in the idle state is $T(1-\rho)$. During this idle time, the service device checks for the presence of requests in the queues before each phase. This leads to the following equality:

$$T\left(1-\sum_{i=1}^{K}\lambda_i t_i\right) = \frac{T}{\overline{T}}\sum_{i=1}^{K} n_i\bar{t}_i^*. \quad (22)$$

Here, $\bar{t}_i^*$ denotes the average time spent checking for the presence of requests for the ith subroutine.

From equation (22), it follows that:

$$\overline{T}_{cycle} = \frac{\sum_{i=1}^{K} n_i\bar{t}_i^*}{1-\sum_{i=1}^{K}\lambda_i t_i}. \quad (23)$$

From the above, the expression for $\bar{t}_{ii}$ can be obtained as:

$$\bar{t}_{ii} = \frac{\sum_{i=1}^{K} n_i\bar{t}_i^*}{n_i(1-\sum_{i=1}^{K}\lambda_i t_i)}. \quad (24)$$

During the period t_{ii}, requests for the i-th phase build up a queue. The distribution function of the workload on the service device for requests that arrive in time t_{ii} is given by

the following expression, which can be obtained using the law of total probability:

$$F_i(t) = \int_0^{\infty} \sum_{m=0}^{[t/t_i]} e^{-\lambda_i t_{ii}} \frac{(\lambda_i t_{ii})^m}{m!} dF(t_{ii}), \qquad (25)$$

where $F(t_{ii})$ is the cumulative distribution function of the random variable t_{ii}.

Suppose t_{ii} has an exponential distribution with mean $\bar{t}_{ii}$. Then expression (25) simplifies to:

$$[F_i(t) = \sum_{m=0}^{[t/t_i]} e^{-\lambda_i \bar{t}_{ii}} \frac{(\lambda_i \bar{t}_{ii})^m}{m!}. \qquad (26)$$

Expression (26) can be given as:

$$F_i(t) = \sum_{m=0}^{\infty} e^{-\lambda_i \bar{t}_{ii}} \frac{(\lambda_i \bar{t}_{ii})^m}{m!} \varphi(t - mt_i), \qquad (27)$$

where $\varphi(t-mt_i)$ is the indicator function that is $\varphi(t-mt_i) = 0$ is true and $m < [t/t_i]$ otherwise?

Denote $P\{t_{0i} < \theta | t > 0\}$ as the conditional cumulative distribution function of the time spent by requests in the queue before the ith phase, given that the initial period of the device's occupancy (the time spent servicing the requests accumulated in the queue during time t_{ii}) is t ($t > 0$). Then the distribution function of the waiting time for service in the iiith phase is given by:

$$F_i(\theta) = \int_0^{\infty} P\{t_{0i} < \theta | t > 0\} dF_i(t). \qquad (28)$$

Using the invariance properties $P\{t_{0i} < \theta | t > 0\}$ of the distribution function with respect to the service discipline for a single-line single-phase system with K incoming Poisson streams and an arbitrary service-time distribution [33], it can be shown that the following expression, obtained in [11], is valid for the considered system:

$$P\{t_{0i} < \theta | t > 0\} = \frac{W_i(\theta - t)}{W_i(\theta)}. \qquad (29)$$

For $t_i = t$, the function $W_i(\theta)$ is given by:

$$W_i(\theta) = \begin{cases} \sum_{r=0}^{[\theta/t_i]} e^{-\lambda_i (r t_i - \theta)} \frac{[\lambda_i (r t_i - \theta)]^r}{r!}, & \text{at } \theta > 0, \\ 0, & \text{at } \theta = 0, \end{cases}$$

where $W(0)$-is an arbitrary coefficient.

Substituting (27) and (29) into (28) and differentiating to t, we get

$$F_i(\theta) - \frac{1}{W_i(\theta)} \sum_{m=0}^{\infty} \int_0^{\infty} W_i(\theta - t) e^{-\lambda_i \bar{t}_{ii}} \frac{(\lambda_i t_{ii})^m}{m!} u'(t - mt_i). \qquad (30)$$

Considering the property of the derivative of the indicator function,

$$\int_{-\infty}^{\infty} f(x) u'(x - z) dx = f(z)$$

Expression (30) can be simplified to

$$F_i(\theta) = \frac{1}{W_i(\theta)} \sum_{m=0}^{\infty} e^{-\lambda_i t_{ii}} \frac{(\lambda_i t_{ii})^m}{m!} W_i(\theta - mt_i). \qquad (31)$$

After a series of transformations, we obtain the following expression for the distribution function of the waiting time for the start of service for requests at the i-th phase:

$$F_i(\theta) = \frac{e^{-\lambda_i t_i} \sum_{z=0}^{\theta/t_i} \sum_{m=0}^{\theta/t_i} e^{-\lambda_i (z-m) t_i} \frac{\lambda_i^{m+z}}{m! z!} [(m+z)t_i - \theta]^z}{\sum_{z=0}^{\theta/t_i} e^{-\lambda_i z t_i} \frac{[\lambda_i (z t_i - \theta)]^z}{z!}}. \qquad (32)$$

Having the expression for the distribution function of the waiting time for the start of service for requests in the i-th phase, we determine the average time a request spends in the queue before the i-th phase $\bar{t}_{0i}$. By definition, and using expressions (20) and (24), we obtain that the quantity $\bar{t}_{0i}$ the following expression gives:

$$\bar{t}_{0i} = \rho_i \bar{t}_{ii}. \qquad (33)$$

Since the total number of requests for all subprograms per cycle T_{cycles} $n_\circ = \sum_{i=1}^{K} n_i$ and the average duration of device occupancy while servicing requests in all phases $\overline{T}$ can be expressed as the sum of the time fractions spent by the device servicing each request queue v_i, it follows that

$$\overline{T} = \sum_{i=1}^{K} n_i v_i.$$

The average time the device spends servicing requests in the i -th phase is given by:

$$\overline{v}_i = \frac{t_i}{1 - \rho_i}. \qquad (34)$$

Substituting (20) and (34) into (33), we finally obtain

$$\bar{t}_{oi} = \frac{\lambda_i t_i}{n_i} \sum_{j=0}^{K} \frac{t_j n_j}{1 - \rho_j}. \qquad (35)$$

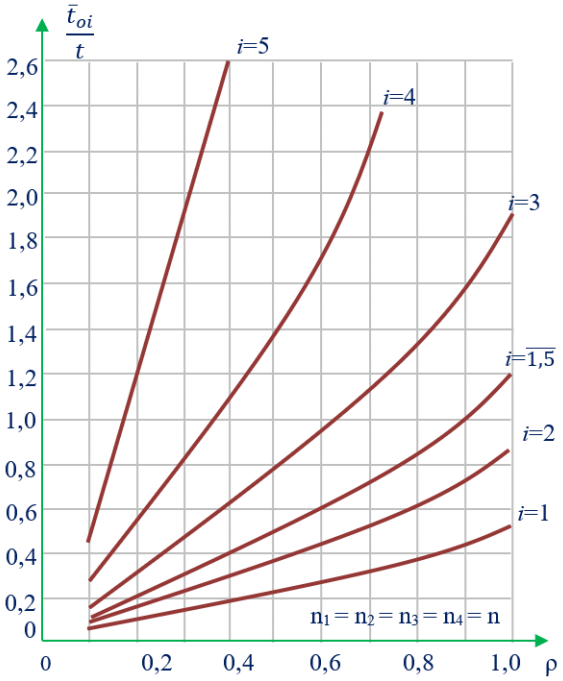

Fig. 7. Increase in device load and waiting time for requests in the queues

The graphs of the normalized average waiting time for the start of service in the i-th phase as a function of the devices load for the second method of processor time allocation are shown in Fig. 6. Calculations were performed using formula (35) for $K = 5$, $\lambda_i = \lambda$, $\tau_i = \tau$, $n_1 = 16$, $n_2 = 8$, $n_3 = 4$, $n_4 = 2$, $n_5 = 1$.

Fig. 7 shows that as the device load increases, the waiting time for queue requests increases rapidly. Additionally, the waiting times for requests in the queues before higher priority phases decrease at the expense of increased waiting times for requests in the queues before lower priority phases. The analysis of the graphs reveals that t_{0i} strongly depends on the total processor load ρ, especially for lower-priority phases.

The average time a request spends in the i-th phase is given by:

$$\overline{\tau}_i = t_i + \frac{\rho_i}{n_i} \sum_{j=1}^{K} n_j \frac{\tau_j}{1-\rho_j}. \tag{36}$$

For equal request arrival intensities at each phase $\lambda_i = \lambda$ and $\tau_i = \tau$, expression (36) simplifies to:

$$\overline{\tau}_i = t \left[1 + \frac{\rho_i n_o}{(1-\rho_i) n_i} \right]. \tag{37}$$

The dependence, constructed using formula (37), is shown in Fig. 8. The calculation was performed for the case $K = 5$, $n_i = 2^{i-1}$. The data from the graph allow us to reproduce the behavior of the curves presented in Fig.7. We analyze the impact of the number of phases in a system of type (6) under the second of the methods considered of processing time allocation on the average waiting time for the start of service in the ith phase.

A. Impact of Processor Load and Priority Levels

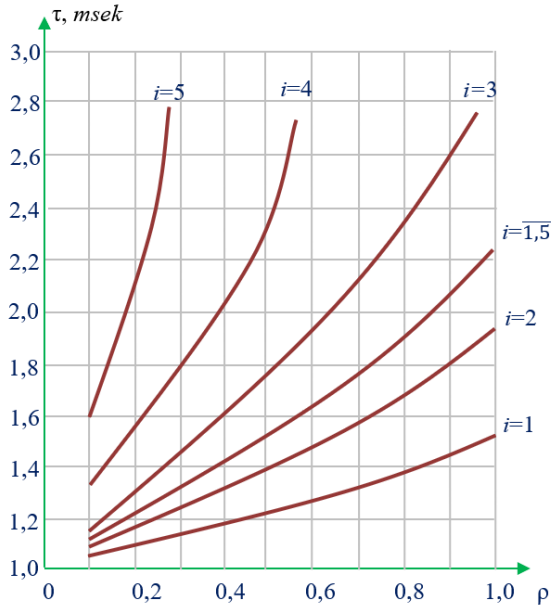

Fig. 8. Graphs of the normalized average waiting time for requests in the queue as a function of the device load

Figure 8 illustrates that as the number of phases increases, lower-priority tasks suffer significantly from queuing delays. This emphasizes the importance of optimizing scheduling strategies in systems with multiple competing subroutine requests.

This section explores how processor load (ρ) and the number of priority levels (K) affect the average waiting time across service phases. As the number of phases increases, requests in lower-priority phases experience significant delays under high device load conditions.

Figures 9–11 show the graphs of the normalized average waiting time for requests in the queue as a function of the device load for systems with 1, 2, 3, 4, and 10 priorities. These graphs indicate that as the overall load on the service device and the number of phases in the system (and, consequently, the number of priorities) increase, the waiting time for the start of service in the queues for higher priority phases increases much more slowly with the total load compared to the waiting time for lower priority phases. Figures 9 and 10 show how queueing time sharply increases for lower-priority phases, especially as processor utilization exceeds 70%. This is a critical insight for embedded systems where timing guarantees are required even for non-critical tasks. Based on the graphs in Figures 9–11, generalized graphs of the dependence of the average normalized waiting time in queues for lower priority phases are constructed according to the level of service device load ρ and the number of priorities (Figure 12). The graphs in Figure 12 show that the extent of the influence of the total number of priorities on the waiting time for requests in the queues significantly depends on the processor load [34]. This influence is minor for small values of $\rho \leq 0$. However, as ρ increases, the waiting times for requests increase dramatically, especially when the number of priorities (phases) is $K \geq 5$.

The sharp increase in the waiting time for requests in the queues for lower priority phases is related to the increased load on the device due to servicing requests at higher priority phases.

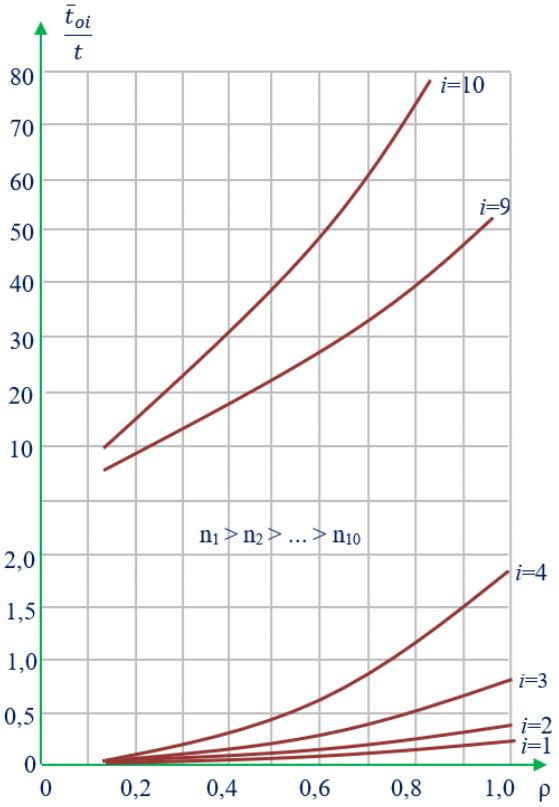

Fig. 9. This graph shows how the average normalized waiting time for different priority levels evolves as overall device load increases in a multiphase queuing system

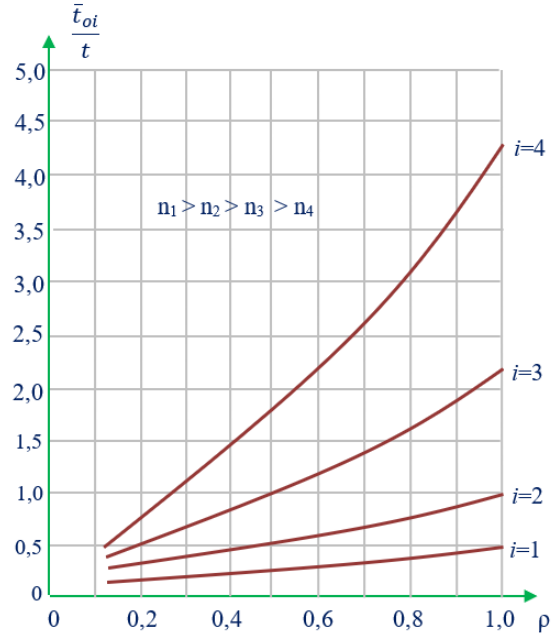

Fig. 10. Illustrates the rapid increase in queueing time for lower-priority phases as device load approaches saturation

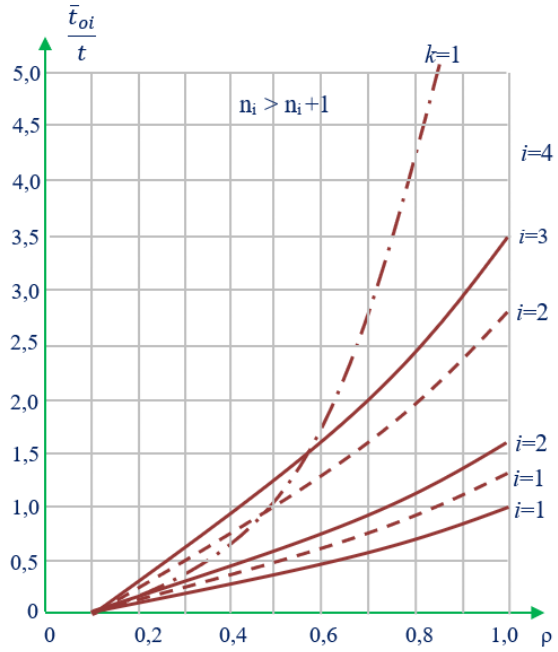

Fig. 11. Busy Period Estimation for the Control Processor – Case A

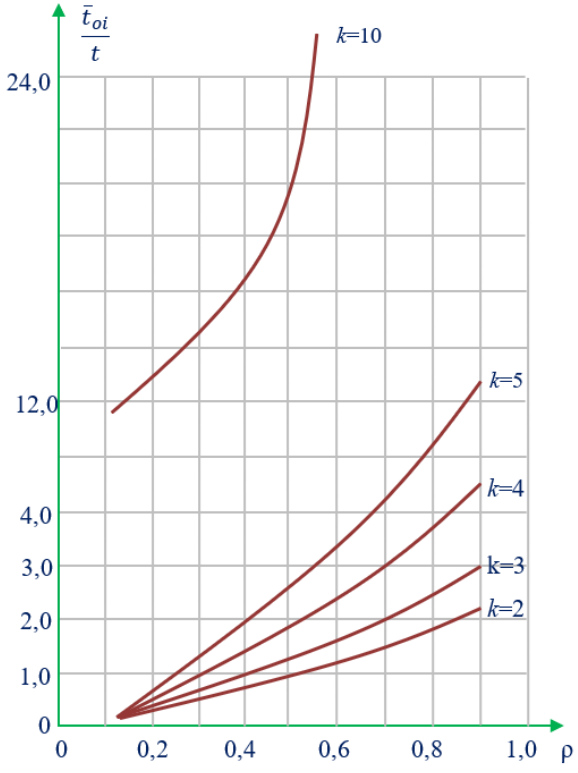

Fig. 12. Effect of Increasing Priority Levels on Waiting Time for Lower-Priority Tasks Under Varying Device Loads-Case B)

To complete the study of the considered model, we determine the average duration of the device's busy period, which is necessary to assess the performance, time, cost efficiency, and throughput of the selected scheduling algorithm. The busy period C is defined as the duration of the time interval from the arrival of a request that finds the device idle to the subsequent

moment when the device becomes idle again after servicing the request. To estimate the average busy period, we divide the time axis into alternating intervals during which the device is either busy or idle. For independent streams, the stationary probability that the device is not processing requests from the i- th stream is $1 - \rho_i (i = \overline{1, K})$, which is invariant to the duration distribution τ_i.

Figure 12 generalizes the behavior across different numbers of priority levels. The results reveal that increasing priority count worsens latency for low-priority phases, especially when $\rho > 0.7$. Thus, limiting excessive prioritization levels may improve overall system fairness.

Consequently, over a very large time interval T, the device will be in the state of not servicing requests from the ith stream for a total time $T(1 - \rho_i)$, which consists of $T\lambda_i(1 - \rho_i)$ non-overlapping busy periods for servicing requests from the i-th stream. The total duration of these various busy periods $T\lambda_i(1-\rho_i)$ equates to $T\rho_i$. Therefore, the average duration of the busy period of the device while servicing all Kth phases is given by:

B. Comparison with Existing Methods

To assess the effectiveness of the proposed execution time estimation models, we compare our approach with several existing methods from the literature.

Jackson's Queueing Model [25] is a classical approach that analyzes multistage queues assuming Poisson arrivals and exponential service times. While mathematically elegant, it fails to account for load-dependent service rates and execution phase prioritization, which are addressed in our proposed method.

Dynamic QoS Management in 5G/6G Networks [28] offers flexible scheduling but primarily focuses on service prioritization in mobile networks, not computational systems. Our model instead targets execution flow in embedded environments.

LLM-based Active Queue Management [29] introduces AI-based solutions for traffic shaping. However, it does not yield closed-form timing expressions nor address multistage program logic.

Credit-Based Network Calculus for Airborne Systems [32] provides precise traffic shaping via calculus but is optimized for communication latency—not processor-bound execution time under stochastic load patterns.

This makes it more suitable for embedded systems and real-time information transmission scenarios, as supporte.

TABLE I
COMPARISON OF EXECUTION TIME ESTIMATION APPROACHES

Method	Multi phase	Closed form	Finite source	Priority modeling	AITS specific
Jackson [25]	No	Yes	No	Limited	No
ML-based [31], [37]	No	No	No	No	No
5G/6G QoS [28]	No	No	No	Yes	No
Proposed	Yes	Yes	Yes	Yes	Yes

As shown in Table I, the proposed method is not intended to replace network-level QoS optimization or data-driven predictors, but rather to provide an analytical framework for structured multi-phase CCR timing in AITS.

C. Device Busy Period and System Throughput

To evaluate system performance holistically, the average busy period of the processor is analyzed. The busy period indicates the total time the device remains active before becoming idle again. This metric is critical for determining throughput and evaluating scheduling efficiency.

$$\bar{C} = \sum_{i=1}^{K} \frac{T\rho_i}{T\lambda_i(1-\rho_i)} = \sum_{i=1}^{K} \frac{t_i}{1-\rho_i}. \tag{38}$$

Assuming that the device spends time sampling requests for service in the ith phase τ_i^*, the complete busy period of the device will be equal to

$$\bar{C}_o = \sum_{i=1}^{K} \frac{\bar{t}_{oi} + t_i + \bar{t}_i^*}{1-\rho_i}. \tag{39}$$

Knowing the numerical value of C_0, the device's throughput can be estimated as follows. Suppose, for example, the processor speed V (instructions/time unit) with an average instruction execution time $\bar{t}$ (time units) and the number of instructions required to process a single request in isolation is m (instructions/request); then the number of requests that the processor can handle per time unit is given by

$$A = \frac{C_o}{mt} \left(\frac{\text{messages}}{\text{unit of time}} \right). \tag{40}$$

Expression (35) is obtained for the case $R = 1$. If m_i streams pass through the ith phase, the quantity t_{0i} is given by

$$\bar{t}_{oi} = \frac{t_i \sum_{j \in m_i} \lambda_j}{n_i} \sum_{g=0}^{K} \frac{t_g n_g}{1 - t_g \sum_{j \in m_i} \lambda_{ij}}. \tag{41}$$

Other characteristics of the model of type (7) can be obtained using formulas (4) and (5). For models of type (7), when $R = 1$, to find an upper bound for the quantity t_{oi}, one can use the expression for the distribution function of the number of requests in each phase of a Markov system [16], [22]. The probability that there are n_K requests at the ith phase of the system is given by

$$P^{(i)}(n_k) = \frac{\delta(n_k)}{\sum_{K=0}^{\infty} \delta(n_k)}, \tag{42}$$

where

$$\delta(n_k) = \begin{cases} (n_k)^{-1}(C_k\rho_k)^{n_k}, & n_k < C_k \\ (C_k)^{-1}(C_k\rho_k)^{C_k}(\rho_k)^{n_k - C_k}, & n_k \geq C_k. \end{cases}$$

Using (42), we obtain the expression for the average number of requests in the queue before the i-th phase and being serviced at the ith phase

$$n_i = \frac{1}{A_i} \left\{ \sum_{r=1}^{C_i - 1} \frac{(C_i\rho_i)^r}{(r-1)!} + \frac{(C_i\rho_i)(\rho_i - C_i\rho_i + C_i)}{C_i!(1-\rho_i)^2} \right\} \tag{43}$$

where

$$A_i = \sum_{r=0}^{C_i} \frac{1}{r_i}(C_i\rho_i)r + \frac{(C_i\rho_i)^{C_i}}{C_i} \cdot \frac{1}{1-\rho_i}.$$

The average number of requests waiting in the queue before the ith phase

$$\overline{L}_{oi} = \frac{(C_i\rho_i)^{C_i}\rho_i}{C_i!(1-\rho_i)^2 A_i}, \tag{44}$$

where A_i – is determined by expression (43). The average waiting time for the request in the queue before the ith phase

$$\bar{t}_{oi} = \frac{\overline{L}_{oi}}{\lambda}. \tag{45}$$

Other characteristics of this model can also be obtained using formulas (4) and (5). For the analysis of models of type (8), it is essential to note that, according to Jackson's theorem [15], a steady-state regime in a multiphase system can be represented as a superposition of the steady-state regimes of individual phases, considered mutually independent. This implies that Cobham's formula [10] for multi-line systems can be used to calculate the characteristics of the system of type (8).

V. Concluding Remarks

This paper has introduced a queueing-theoretic framework for estimating the execution time of computer programs in Automated Information Transmission Systems (AITS). Two practically relevant categories of control computing resources (CCR) were identified: Type I resources with multistage program execution and strict timing constraints, and Type II resources characterized by finite source devices and unreliable components. For both categories, we derived closed-form analytical expressions for key temporal metrics, including average waiting time, execution delay, queue length, and processor busy period, under fixed-priority and cyclic weighted scheduling disciplines. The analytical models were validated through simulation experiments, which confirmed that the proposed formulas capture the expected behavior of multiphase CCRs under varying processor loads and priority configurations. In particular, the results demonstrate how increasing the number of phases and priority levels leads to a sharp growth in waiting time for lower-priority tasks as utilization approaches saturation, while high-priority tasks remain relatively protected. These insights are directly applicable to the design and configuration of real-time AITS controllers [35], in line with recent execution-time analysis and priority-based scheduling methods in real-time and embedded systems, where the choice of scheduling discipline, number of priority classes, and acceptable processor load must balance responsiveness for critical functions against fairness for non-critical tasks [10], [24], [26]. Execution time estimation in real-time, embedded, and distributed systems has been intensively studied over the last few decades, with several mainstream directions emerging [30], [31], [35], [36]. From a practical standpoint, the derived expressions for busy period and throughput can be used as design tools to dimension processor resources and to estimate how many control tasks per time unit can be reliably executed, given a specific instruction rate and program complexity. This makes the framework suitable as an analytical complement to measurement-based or simulation-based performance evaluation in embedded AITS deployments [37]. The present work has several limitations that suggest directions for future research. First, the models largely rely on Poisson arrival processes and deterministic service times, which lead to conservative (overestimated) timing characteristics but may not fully capture bursty or correlated traffic patterns observed in modern communication-intensive control systems. Extending the framework to more general arrival and service-time distributions, or to hybrid analytical–measurement-based schemes, is a promising direction. Second, integrating the proposed CCR models with network-level scheduling and QoS mechanisms in 5G/6G or digital twin architectures would allow joint optimization [15], [24], [28], [32]. Future research will focus on extending the proposed models to hybrid analytical–data-driven frameworks, integrating adaptive scheduling mechanisms and learning-based parameter estimation to enhance robustness under non-stationary workloads.

References

[1] M. Kenyeres, J. Kenyeres and R. Burget, "Evaluation of natural robustness of best constant weights to random communication breakdowns," *J. Commun. Softw. Syst.*, vol. 14, no. 3, pp. 201–210, Sept. 2018, doi: 10.24138/jcomss.v14i3.487.

[2] P. Sousa, V. Pereira, P. Cortez, M. Rio and M. Rocha, "A framework for improving routing configurations using multi-objective optimization mechanisms," *J. Commun. Softw. Syst.*, vol. 12, no. 3, pp. 145–156, Sept. 2017, doi: 10.24138/jcomss.v12i3.79.

[3] C. H. Ng and B. H. Soong, *Queueing Modelling Fundamentals*, 2nd ed. Hoboken, NJ, USA: Wiley, 2008. ISBN: 978-0-470-99467-2.

[4] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data," *IEEE J. Sel. Areas Commun.*, vol. 4, no. 6, pp. 833—845, Sept. 1986, doi: 10.1109/JSAC.1986.1146373.

[5] M. Mitzenmacher and R. Shahout. "Queueing, Predictions, and Large Language Models: Challenges and Open Problems," *Stochastic Systems*, vol. 15, no. 3, pp. 195–219, July 2025, doi: 10.1287/stsy.2025.0106.

[6] M. Mirzaeva and A. Sulaymonov, "One of the methods for assessing the reliability of a communication network," in *International Conference on Information Science and Communications Technologies (ICISCT)*, Tashkent, Uzbekistan, 2019, pp. 1–4.

[7] M. Mirzaeva and M. Sobirov, "Estimates of efficiency and control methods of communication network functioning," *International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE)*, vol. 9, no. 4, pp. 5736–5740, 2020.

[8] N. Okafor, B. Lusch and V. Vishwanath, Queue wait time prediction in high performance computing (HPC) systems," *The Journal of Supercomputing*, vol. 82, no. 1, pp. 90–119, Jan. 2026, doi: 10.1007/s11227-025-08221-7.

[9] M. Mirzaeva, M. Sobirov and M. Bafoyev, "Study of neural networks in telecommunication systems," in *Proc. ICISCT*, Tashkent, Uzbekistan, 2021, pp. 1–4.

[10] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, New York, 2013. ISBN: 978-1-107-02750-3.

[11] A. Tamuli, D. Das, V. Deepthi, A. Choudhury, D. Bora, B. Patowari and S. Mahanta, Estimation of performance measures in a novel M/M/1 queueing model with reverse balking: A simulation-based approach," *Results in Control and Optimization*, vol. 20, pp. 100590, June 2025, doi: 10.1016/j.rico.2025.100590.

[12] R.O. Schoeneich, M. Golanski, M. Kucharski, M. Franciszkiewicz and D. Zgid, "The channel for hidden data transmission in WSN," *International Journal of Electronics and Telecommunications*, vol. 63, no. 2, pp. 209–216, 2017, doi: 10.1515/eletel-2017-0028.

[13] P. Annabel and K. Murugan. "Energy-Efficient Quorum-Based MAC Protocol for Wireless Sensor Networks". ETRI Journal, vol. 37, no. 3, 2015.

[14] T. Rault, A. Bouabdallah and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Comput. Netw.*, vol. 67, pp. 104–122, July 2014, doi: 10.1016/j.comnet.2014.03.027.

[15] M. Ebrahimabadi, J. Bahrami and M. Younis, "Digital twin integrity protection in distributed control systems," in *Proc. 21st IEEE Int. Conf. Embedded & Ubiquitous Comput.(EUC)*, 2024, pp. 1–8, doi: 10.1109/CCNC51664.2024.10454732.

[16] A. Coluccia and G. Notarstefano, "A Bayesian framework for distributed estimation of arrival rates in asynchronous networks," *IEEE Trans. Signal Process.*, vol. 64, no. 15, pp. 3984–3996, Aug. 2016, doi: 10.1109/TSP.2016.2557313.

[17] I.D. Schizas, A. Ribeiro and G.B. Giannakis, "Consensus in ad hoc WSNs with noisy links—Part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process.*, vol. 56, no. 1, pp. 350–364, Jan. 2008, doi: 10.1109/TSP.2007.906734.

[18] D. C. Vasiliadis,et al., "Performance Tuning of Dual-priority Delta Networks through Queuing Scheduling Disciplines," *J. Commun. Softw. Syst.*, vol. 9, no. 4, pp. 222–235, Dec. 2013. doi: 10.24138/jcomss.v9i4.143

[19] G.S. Seyboth, D.V. Dimarogonas and K.H. Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, vol. 49, no. 1, pp. 245–252 (2013). 10.1007/978-0-85729-033-5(3)

[20] C. Nowzari and J. Cortés, "Zeno-free, distributed event-triggered communication and control for multi-agent average consensus," in Proc. ACC 2014, Portland, OR, United States, 2014, pp. 2148–2153. doi: 10.1109/ACC.2014.6859495

[21] S. Vanka, V. Gupta, and M. Haenggi. "Power-delay analysis of consensus algorithms on wireless networks with interference. " *International Journal of Systems, Control and Communications*, vol.2, no. 1–3, pp. 256—274 (2010). doi: 10.1504/IJSCC.2010.031166

[22] F. Garin and L. Schenato. "A survey on distributed estimation and control applications using linear consensus algorithms." *Lecture Notes in Control and Inform. Sci.*,vol.406, pp. 75-107, (2010). doi: 10.1007/978-0-85729-033-5-3

[23] S. Kar, R. Tandon, H.V. Poor, and S. Cui. "Distributed detection in noisy sensor networks." *ISIT*, St. Petersburg, Russia, pp.2856 —2860. (2011) doi: 10.1109/ISIT.2011.6034097.

[24] M. Iqbal, M. U. Shafiq, S. Khan, Obaidullah, S. Alahmari and Z. Ullah, "Enhancing task execution: a dual-layer approach with multi-queue adaptive priority scheduling," *PeerJ Comput. Sci.*, vol. 10, p. e2531, Dec. 2024, doi: 10.7717/peerj-cs.2531.

[25] F. A. Silva, T. A. Nguyen, I. Fé, C. Brito and D. Min, "Performance Evaluation of an Internet of Healthcare Things for Medical Monitoring Using M/M/c/K Queuing Models," *IEEE Access*, vol. 9, pp. 56345–56358, 2021, doi: 10.1109/ACCESS.2021.3072126.

[26] M. Yin, M. Yan, Y. Guo and M. Liu, "Analysis of a Pre-Emptive Two-Priority Queuing System with Impatient Customers and Heterogeneous Servers," *Mathematics*, vol. 11, no. 18, pp. 3878, Sept. 2023, doi: 10.3390/math11183878.

[27] J. Anand and B. Karthikeyan, Dynamic priority-based task scheduling and adaptive resource allocation algorithms for efficient edge computing in healthcare systems," *Results in Engineering*, vol. 25, pp. 104342, Feb. 2025, doi: 10.1016/j.rineng.2025.104342.

[28] P. D. Bojović, M. Radojević, A. Kostić, M. Popović, and D. Mitić, "Dynamic QoS Management for a Flexible 5G/6G Network Core: A Step Toward a Higher Programmability," *Sensors*, vol. 22, no. 8, p. 2849, 2022, doi: 10.3390/s22082849.

[29] D. Efrosinin, V. Vishnevsky and N. Stepanova, "Optimal Scheduling in General Multi-Queue System by Combining Simulation and Neural Network Techniques," *Sensors*, vol. 23, no. 12, pp. 5479, June 2023, doi: 10.3390/s23125479.

[30] A. P. Singh and A. Bhagat, "Performance analysis of unreliable finite queue with tri-control policies (N, F, q) and repairs," *Journal of Control and Decision*, pp. 1–18, Dec. 2025, doi: 10.1080/23307706.2025.2589381.

[31] S. S. Sanga and Nidhi, "Machine learning to manage retrial queueing system M/G/1/N under F-policy with discouraged customers and fuzzy costs and its implementation in medical centers," *Journal of Computational and Applied Mathematics*, vol. 473, pp. 116843, 2026, doi: 10.1016/j.cam.2025.116843.

[32] R. Yan, Q. Li and H. Xiong, "Optimizing Traffic Management in Airborne Power Line Communication Networks: A Credit-Based Shaping Approach Using Network Calculus." *IEEE Transactions on Network and Service Management*, vol. 22, no. 2, pp. 1437–1449, Apr. 2025, doi: 10.1109/TNSM.2025.3529871.

[33] B. S and M. C. S, "Unreliable M[X]/G(P1,P2)/1 feedback retrial queues with combined working vacation," *Heliyon*, vol. 10, no. 17, pp. e36778, Aug. 2024, doi: 10.1016/j.heliyon.2024.e36778.

[34] X. Yang, Y. Zhang, B. Wang, and X. J. Li, "Performance Analysis and Cost Optimization of the M/M/1/N Queueing System with Working Vacation and Working Breakdown," *Mathematics*, vol. 13, no. 18, pp. 2980, Sept. 2025, doi: 10.3390/math13182980.

[35] V. Kozyrev, "Estimation of the execution time in real-time systems," *Program. Comput. Software.*, vol. 42, no. 1, pp. 41–48, 2016.

[36] P. Porjaroenphan, S. Bevinakoppa and P. Zeephongsekul, "A method for estimating the execution time of a parallel task on a grid node," in *Proc. European Grid Conf.*, 2005, pp. 226–236.

[37] K. Taunk, S. De, S.Verma, and A. Swetapadma. "A brief review of nearest neighbor algorithm for learning and classification." *International Conference on Intelligent Computing and Control Systems (ICCS)*, pp. 1255—1260 (2019). doi: 10.1109/ICCS45141.2019.9065747

Mirzaeva Malika received the Ph.D. degree in System Analysis, Management, and Information Processing. She is an associate professor at "Tashkent Institute of Irrigation and Agricultural Mechanization Engineers" National Research University, Uzbekistan, and Tashkent International University. In the past, her research interests and her Ph.D. dissertation were focused on models and algorithms of decision-making in management and multilevel hierarchical communication networks. Nowadays, she is working on artificial intelligence and digital technologies. The research topics include telecommunication systems, QoS, embedded systems, network monitoring and intrusion detection in industrial networks, and management of data communication.

Suleymanov Anvar is a DSc. at the "Tashkent Institute of Irrigation and Agricultural Mechanization Engineers" National Research University, Uzbekistan. He completed his DSc degree in 1992, specializing in theoretical foundations of informatics. His research interests include programming, mathematics, and quantum computing. He is the author of more than 30 articles

Mamatov Narzullo is a full professor and Head of the Digital Technologies and Artificial Intelligence Department at "Tashkent Institute of Irrigation and Agricultural Mechanization Engineers" National Research University, Uzbekistan. His research interests include image processing, voice recognition, AI in healthcare, AI in finance, AI in robotics, knowledge representation and reasoning, quantum computing and AI, AI and edge computing, and AI development tools and frameworks.