# A Comprehensive System Architecture Model for Telematics Systems of Multimodal Transport Hubs

NATALIA MAMEDOVA, MIKHAIL AFANASEV, ARKADIY URINTSOV,
SERGEY HARITONOV
Basic Department of Digital Economy,
Plekhanov Russian University of Economics,
36, Stremyanny Lane, Moscow, 117997,
RUSSIA

*Abstract:* - This paper presents a theoretical model of a system architecture designed to address integration challenges in multimodal hub telematics. Traditional design methods, with their rigid "function-device" sequence, fail to handle heterogeneous data from sources such as AIS and RFID. The proposed model replaces fragmented ("patchwork") solutions by coordinating edge nodes for real-time sensor data preprocessing with scalable cloud resources for batch analytics. The model's core components include dynamic gateway deployment based on NFV and key orchestration elements, such as a load balancer engineered to handle cargo flow spikes. The developed model serves as a foundation for building reference architectures tailored to the specific requirements of multimodal hub telematics. The paper outlines the model's constituents, its scope of application for various hub types, and its implementation stages.

*Key-Words:* - theoretical model, distributed computing, transport logistics, computational architecture, telematics, multimodal hub, resource orchestration, software-defined networking.

Received: April 19, 2025. Revised: July 11, 2025. Accepted: August 19, 2025. Published: February 18, 2026.

## 1 Introduction

The modern economic environment demonstrates the intensification of competition between companies that are actively developing distributed and peripheral computing models. In the field of transport logistics, if we look at our experience, this trend is particularly pronounced: there are stricter requirements for data security and control of their lifecycle, as well as a constant increase in the volume of information that requires processing directly at the place of origin – on the periphery. It seems to us that these trends, reinforced by recent global challenges, are actively stimulating the demand for decentralization of logistics processes. It should be noted that the development of network infrastructure is often uneven, and the existing components of the system architecture do not always meet the growing needs for modern network functions.

In an effort to adapt to the dynamic environment of data generation, processing, storage and transmission, enterprises face the need to optimize their system architecture. However, as our analysis has shown, developing your own solutions remains a difficult and expensive task. The problem is compounded by the lack of standard reference models that could integrate best practices and optimization methods, considering the specifics of telematics systems. In this context, the present study aims to develop an integrated system architecture model for a multimodal transport and logistics center. The proposed model combines network topology elements for telematics devices available on the Russian market and meets strict requirements for security, performance and flexibility.

We determine the scientific relevance of the work through the need to analyze the key factors shaping the architecture for warehouse logistics enterprises:

1. Conditions for the integration of cloud and peripheral layers in system architecture modeling.
2. A variety of mechanisms for distributing computing load and data transmission channels between nodes.
3. Ensuring compatibility of hardware and software solutions is of particular importance.

From a practical point of view, the significance of our work lies in the possibility of using the model to develop and optimize the system architecture of logistics enterprises. The results obtained can serve as a basis for substantiating decisions on IT and

technical equipment of warehouse infrastructure, ensuring a balance between industry requirements and market offers.

The scientific novelty of our approach consists in complex modeling, which combines hybrid cloud services and distributed peripheral computing, as well as solves the problems of service integration, considers the requirements for data storage systems and the limitations imposed by Russian hardware and software solutions with pre-installed integration.

If we consider the conceptual framework, then a multimodal transport hub is a point of interaction between two or more modes of transport, acting as a key element of the route network, where their technological interaction is provided by a complex of transport devices, facilities and systems, [1], [2]. The organizational and legal core of such a hub is a multimodal transport and logistics center, which we consider to be a critically important element of the regional transport infrastructure. In our opinion, the operational efficiency of the MTLC, which ensures the coordinated interaction of all participants in logistics processes, directly depends on the correctness of the design of its system architecture – this principle, we believe, is applicable to centers of any scale.

## 2 Materials and Methods

The conducted analysis revealed that traditional Our analysis of traditional methods for building computational and engineering architectures in MTLC telematics systems revealed their fundamental constraint: a rigid adherence to the "function-device/service" principle. We observed that this paradigm creates a significant operational bottleneck – every time there is a need to increase computing power or introduce a new service. It inevitably leads to costly overhauls of the network topology or the installation of specialized hardware. As our work progressed, we identified three core limitations that plague this traditional approach:

a) unpredictable fluctuations in network bandwidth demands;
b) compatibility issues between heterogeneous equipment, necessitating lengthy and expensive preliminary testing;
c) significant hurdles in integrating multi-vendor solutions, which in turn drive up operational costs.

In practice, we see that these limitations rarely arise in isolation. They are usually intertwined, creating complex optimization problems. This often forces logistics sector organizations to make customized, fragmented decisions. Although such approaches can minimize initial costs, our results show that they ultimately lead to inefficient and expensive architectural upgrades in the future.

The main goal of this project was to find a way to overcome the limitations we described. In this study, we have developed a comprehensive model of the system architecture of telematics systems in multimodal transport hubs. This model facilitates the integration of both physical and virtual network devices with predefined functions. We believe that this will help effectively bridge the gap between the capabilities of the hardware platform and the growing demand for modern network services. From an operational standpoint, the key advantages of our model are its ability to assist in:

a) designing a functional network topology that considers the capabilities of available equipment;
b) precisely calculating the optimal load on devices;
c) substantiating requirements for the deployment of new devices;
d) formalizing parameters for the evaluation of new equipment.

The results of our research show that the proposed model has an advantage over traditional approaches, especially in the field of operational efficiency, cost reduction and scalability of the telematics system.

### 2.1 Scope and Nature of the Problem to be Solved

Our research confirms that the transport system of the Moscow agglomeration remains one of the most efficient in the world. At the same time, we observe how modern trends in the Russian IT industry create unique conditions for the implementation of hardware and software solutions that meet the new requirements of the IT infrastructure of warehouse logistics. According to our estimates, by 2025, digitalization of production can increase Russia's annual GDP by 1.3-4.1 trillion rubles, which is consistent with the conclusions of the McKinsey Global Institute on the potential increase in productivity in logistics by 45-55% due to digital technologies. This growth, in our opinion, is driven by the growing demand for scalable computing and data storage in logistics operations.

Multimodal transport hubs are fundamental elements of regional systems, and their effectiveness directly depends on a well-designed system architecture that includes both physical network devices and multi-level computing services (cloud and peripheral). In our work, transport telematics, based on the application of mathematics and

computer science, is considered as a key tool for designing such an architecture of a multimodal node.

Our market analysis showed steady growth of the hybrid cloud and distributed computing segment in Russia over the past five years, which coincided with an increase in the load on traditional computing nodes. Although the integration of additional hybrid and distributed services offers a tactical solution, we concluded that such a fragmented approach creates serious problems for the integration of services, hardware and storage systems, which leads to reduced efficiency due to unpredictable dependencies of devices and services. The theoretical model we have developed bridges this gap by offering evidence-based support for technical solutions at the cloud and peripheral levels for multimodal nodes with proven applicability in the development of digital twins.

Distributed computing, which has remained an active area of research for more than six decades, solves ongoing problems such as scalability and data distribution, [3], [4]. Fundamental research, in our understanding, is aimed at solving practical problems of network operation, including increasing bandwidth requirements. Companies such as Sun Microsystems Inc. have made historical contributions, while modern innovations are being developed by players such as Yandex and IBM Corp. In addition to commercial applications, we believe that this area of research is critically important for creating fault-tolerant systems in scientific projects and promoting the concept of a distributed computing continuum, [5].

Our analysis of resources for optimizing architectures for complex systems, such as multimodal nodes, allowed us to confirm the following points.

Industry 4.0 requires end-to-end digitalization of logistics chains into a single ecosystem, [6], where multimodal nodes serve as the main nodes, [7]. Among the dominant trends in supply chains, we highlight the absence of intermediaries, comprehensive resource sharing and the introduction of digital technologies (RFID, IoT, blockchain, etc.), [8], [9], [10]. In our opinion, the development of the model should correspond to the cloud production paradigm, [11] and explore the application of distributed computing in transport management, [12], [13], [14].

Logistics efficiency assessment should be considered in the context of methodologies that require flexibility, fault tolerance, and real-time data exchange, where competitive advantages arise as a result of structural changes caused by digitalization, [15]. Empirical testing, as our experience has shown, confirms that the digitalization strategies of logistics companies have a modular structure, while data processing technologies are a priority module that allows them to adapt to the market, [16].

## 2.2 Suggested Methods and Approaches to Solve the Problem

We formalized the theoretical model through five key methodological approaches, each chosen to address a specific aspect of the architectural challenge.

First, we turned to numerical optimization methods, framing the core problem as the minimization of an objective function under a set of constraints. Our process involved identifying key architectural elements and mapping the dependencies between devices and computing services using mathematical relations and graphs. During our evaluation of available tools, the Hooke-Jeeves direct search method stood out for its derivative-free operation, which we empirically validated. This work, however, surfaced significant computational challenges: we observed a general insensitivity to specific project variables, such as the distinction between peak and average load, and a noticeable bias toward prioritizing software solutions over hardware.

Second, we employed analytical modeling to verify the model's overall adequacy. By using "soft" modeling techniques, we were able to derive qualitative conclusions from differential equations. This approach proved invaluable for identifying broader development patterns and analyzing how both external and internal factors influence the system architecture.

Third, we applied Network Functions Virtualization (NFV) technology, which served as the backbone for meeting several key harmonization criteria. Our implementation focused on achieving a clear separation between hardware and software, enabling the automated and scalable deployment of network functions, and establishing granular operational control through precise monitoring.

Fourth, conceptual mathematical modeling helped us establish the fundamental solutions. This involved a three-part process: defining the composition, structure, and objective function of the control system; describing potential system states and relevant performance metrics; and formalizing the relationships between inputs, states, and outputs alongside their implementation requirements.

Finally, to ground the model in reality, we constructed functional dependencies between physical devices and cloud/edge services. For this,

we relied heavily on data extracted from technological observation maps of actual multimodal transportation hubs.

# 3 Results

Our research has led us to two key conclusions. Firstly, empirical verification has convincingly confirmed that individual solutions and fragmented optimization are methodologically unacceptable for building multimodal transport and logistics architectures. Our proposed model establishes a set of immutable design principles specifically designed to avoid this trap by creating a viable and consistent architectural structure. Secondly, we argue that the model can be considered a reference standard because it successfully combines four main components: 1) Resource optimization methods using Hooke-Jeeves algorithms and API-level constraints. They provide parameterized load balancing in accordance with GOST standards, forming the mathematical core of our model. 2) Principles of network mapping, formalized using NFV detection and adjacency matrices. They clearly define the dependencies between the device and the service, providing the spatial analysis needed for predictive modeling. 3) An algorithm for resource administration equipped with multi-purpose optimization cores. This component implements real-time management of virtual resources using Internet of Things telemetry, thereby integrating self-regulation capabilities directly into the system. 4) A finite element model translated using Project Studio CS and calibrated in accordance with GOST standards. This ensures full compatibility with the digital twin for comprehensive scenario testing and refinement of the executed project.

Quantitative analysis of the spread of defects revealed a striking contrast: solutions for fragmented optimization showed a significantly higher number of integration defects (62% ± 8.1) and a longer average recovery time (22.7 hours) compared to solutions for individual software architecture. Collectively, our results show that when developing custom architectures, exponential technical debt usually arises due to non-linear integration costs and vendor dependence. Conversely, we found that fragmented optimization is the main cause of system instability, manifested in security breaches at integration boundaries and load balancing failures during peak loads.

Therefore, our model defines the following criteria for the reference architecture:

a) hardware-independent control plane;
b) support for solutions based on national standards;
c) resource consumption in seconds (scaling delay < 0.8 s);
d) Compatibility with digital counterparts (BIM Level 2 LOD 400+).

The theoretical model makes it possible to realize several key architectural achievements. Most notably, it replaces the traditional sequential "function-device-service" approach with a separate resource organization. This fundamental change allows for dynamic hardware scaling using NFV-managed hypervisors and ensures application portability between different vendors, an opportunity that we have successfully tested when modeling migration between Yandex and Rostelecom VNF cloud infrastructures.

# 4 Discussion

## 4.1 Parameterized Load-Balancing Algorithms

Parameterized load-balancing algorithms enabling hardware scaling and control application routing were developed through a zero-order optimization method (Hooke-Jeeves) applied to 5 simulated load scenarios.

The system state is formalized by the vector $S = [R_p, R_v, C, T, P]$,

where:

$R_p$ - vector of available physical resources (CPU cores, RAM GB, Storage IOPS/BW, Network BW).

$R_v$ - vector of the current state of virtual resources (VMs, vStorage, vChannels).

$C$ - vector of constraints (power consumption, temperature, licenses).

$T$ - vector of telemetry data (load, latencies, errors).

$P$ - vector of configuration parameters (task priorities, CII (Critical Information Infrastructure) service classes, SLA).

Before calculating the objective function, deviations from SLA metrics must be determined. In practice, SLAs for different MTLC services differ; for instance, customs transactions are more critical than video analytics. However, the final decision on which metrics to calculate deviations for rests with the system architect.

The logic for calculating the objective function for metrics such as critical transaction latency, network channel jitter, API/transaction error rate,

resource utilization, and hard constraint violations is as follows.

The objective function $F(S)$ minimizes the weighted sum of deviations from the required SLAs:

$$F(S) = w_1 \times L(S) + w_2 \times J(S) + w_3 \times E(S) + w_4 \times U(S) + w_5 \times C(S)$$

where:

$L(S)$ - aggregated latency of critical transactions.

$J(S)$ - network channel jitter.

$E(S)$ - API/transaction error rate.

$U(S)$ - resource utilization imbalance (CPU, RAM, BW, IO).

$C(S)$ - penalty for violating hard constraints (security requirements, power, temperature).

Deviation values are not arbitrary but are calculated by the algorithm based on actual telemetry. Their allowable limits are defined in the SLAs and are input parameters for the optimization. Our understanding of SLA standards considers industry specifics, but it is not a definitive standard – hypothetically, the set of SLAs can be personalized.

The weights $w_i$ are dynamically adjusted based on task priorities within the range [0.0 to 1.0] and the CII object category [no category, 3rd, 2nd, 1st]. For example, for a 1st-category CII object, transaction latency $L(S)$ is more critical than resource utilization imbalance $U(S)$. The weights $w_i$ regulate the importance of each deviation, while the hard constraints in $C(S)$ prevent exceeding the defined limits. The adaptability of the algorithm's control loop is manifested in the dynamic weight adjustment mechanism: $w_i$ change when the CII category changes. For instance, when switching to an emergency mode (1st CII category), the weight for temperature ($w_5$) should be increased.

The Hooke-Jeeves method iteratively searches for the minimum of $F(S)$ in the space of control parameters defined by a vector $X$, which determines resource allocation.

To align with the requirements of Russian jurisdiction, we first verified software solution compliance at the API level within our test environments. Our team also validated the import substitution pathway by working through hardware abstraction levels. As a result, the foundational mathematical framework we developed enables dynamic resource allocation. It directly tackles the "function-device/service" sequence constraint by effectively decoupling functions from the underlying physical hardware.

In this work, we focused on parameterized load-balancing algorithms to solve a core architectural limitation: the traditional rigid link between a function and a specific device or service, which historically demanded hardware reconfiguration for any functional modification. We addressed this by creating a model where key attributes (such as hardware capabilities (CPU cores, memory bandwidth, I/O performance) and network features (routing protocols, security policies)) were abstracted into a set of tunable parameters instead of remaining as fixed pairs. We selected the Hooke-Jeeves zeroth-order optimization method for three principal reasons: its derivative-free nature (essential for dealing with discontinuous objective functions like latency-jitter tradeoffs), its compatibility with black-box hardware simulators, and its demonstrated convergence in high-dimensional parameter spaces.

Our work was guided by the following key technical objectives: The model's features must be adaptable for subsequent deployment of parameter sets onto the target hardware platform. All software and hardware constraints are defined as optimization boundaries. For dynamic scaling, the system must perform resource reallocation in under 15 seconds during a 200% load spike. The targeted level of architectural partitioning aims for the complete elimination of bottlenecks in the function-device sequence.

To rigorously test our approach, we implemented 5 simulated workload scenarios with their specific boundary conditions in the GNS3 Network Simulator. The detailed results are available in a separate whitepaper and leverage industry-standard tools and metrics: 1) A Black Friday transaction peak simulation modeled a 300% transaction increase, with boundary conditions ensuring an end-to-end latency of ≤150ms (RFC 2544), a transaction success rate of ≥99.99% (SLA), and an API error rate of <0.005% (ISO/IEC 25010-2023). 2) A customs clearance delay simulation emulated a 24-hour document processing halt, requiring transaction retention for ≥72 hours (ISO 16363:2025) and system state recovery within ≤45 seconds once service resumed. 3) A thermal throttling simulation recreated data center cooling failure in Project Studio CS, limiting performance degradation to a maximum of ≤15% at an ambient temperature of 85°C. 4) A cross-region migration test evaluated VM transfers between edge cloud tiers under network jitter of 200ms, requiring that service disruption remained below 800ms. 5) A cold storage activation test involved the rapid retrieval of historical IoT data, demanding a throughput of

≥5GB/s and a decompression latency of ≤10ms per file.

This suite of scenarios exposed our parameter space to extreme operational loads while confirming that all deterministic constraints were satisfied. The resulting dataset validated the algorithm's robustness against real-world transportation logistics demands and helped quantify performance ranges for safe, SLA-compliant deployments.

A crucial part of our design was building jurisdictional requirements directly into the optimization loop as invariant constraints. This integration provides continuous API-level compliance checking during solution iterations, ensuring outputs meet technical specifications without needing external validation. We implemented hardware abstraction layers as parameter mapping interfaces, which allow identical network functions (for instance, traffic shaping) to be executed on different hardware by translating universal algorithmic parameters into vendor-specific command sets via dynamically loaded drivers.

Naturally, this abstraction mechanism demands thorough testing of system operation, performance, and hardware compatibility on a physical platform. For example, we verified that a set of load-balancing parameters optimized for brand X switches delivered identical QoS metrics when mapped to brand Y hardware through our abstraction layer. A key goal moving forward is to confirm the long-term functional equivalence of this mechanism across different hardware platforms (Eltex, Biforcom Tek). The final design eliminates the function-device/service sequence constraint by separating functions from physical hardware: network functions become portable feature sets deployable on any compatible hardware, and hardware resources are treated as measurable capacity pools (e.g., "2.4Mpps packet processing" instead of "Model Z router").

We achieve dynamic resource allocation through continuous iterative optimization cycles. In these cycles, the Hooke-Jeeves algorithm fine-tunes feature sets based on live telemetry data and scales virtualized functions across hardware resources – all without physical reconfiguration. This shifts system architecture design from a sequential mapping of functions to specific devices to a parallel mapping of features to available capacities. The tangible outcome of deploying these feature sets onto a hardware platform is a dramatic reduction in resource provisioning latency (with average values dropping from 10 days to just 40 minutes). Ultimately, our approach fundamentally transforms resource allocation architecture from a static binding process to an adaptive parameter mapping one, simultaneously overcoming a major architectural limitation and embedding jurisdictional compliance at the algorithmic level.

## 4.2 Network Mapping Principles

We succeeded in mathematically formalizing the dependencies between devices and services across heterogeneous infrastructure by creating topological maps based on SDN network discovery protocols commonly used in NFV environments. In our understanding, a topological map is a visual representation of the physical connections between devices in an industrial network. We empirically modeled functional connections using adjacency matrices, which became computational bridges between physical devices and cloud services ($(t) = f[X(t)]$). This approach allowed us to simulate "what-if" scenarios, such as equipment failures or demand spikes, before they occur in the live system.

In the course of our work, we created and processed several graphs to visually depict the structure of connections between network equipment. The analysis was based on 78 hours of operational data collected from a pilot multimodal transport hub. The resulting mathematical formalization laid the foundation for a theoretical model capable of predicting the behavior of the system architecture under various conditions.

We developed the topology mapping methodology to solve the critical problem of unpredictable dependencies between devices and services in distributed logistics architectures. In practice, we have repeatedly observed how unstructured connections between physical equipment and cloud/edge services increase the risk of system instability. Historically, configurations of physical devices in industrial networks (switches, sensors, gateways) have exhibited hidden interdependencies with software services, leading to cascading failures during scaling.

Although IP address scanning is traditionally used to discover and identify all devices on a network, our experience showed that this method fails to reveal functional relationships. For instance, it cannot answer the question of how a temperature sensor data flow influences container routing decision.

To tackle this problem, we took the following steps. We deployed SDN topology discovery protocols from NFV environments because they reveal data link layer (Layer 2) topology and service

binding metadata. As a criterion for network protocol configuration, we defined the dependence of an MQTT broker on the latency threshold of a specific switch port. For distributing computing resources, we assigned a GPU node in the cloud environment to process requests to a specific endpoint of the video analytics service API.

After selecting the network protocols, we faced the task of understanding how devices and services actually interact within the infrastructure. We concluded that while protocols define the rules for data exchange, they often do not clearly show the functional dependencies between system components. The essence of topological transformation, from our perspective, is to obtain a formalized and visual map of interactions that reflects not only physical connections but also real logical dependencies.

The raw protocol data constitutes unprocessed information about network connections, events, requests, and responses between devices and services. We view topological transformation as the process of structuring and visualizing this information using graph theory methods.

In our model, graph nodes correspond to individual devices or services on the network, while edges reflect functional dependencies between nodes. Among the constructed edges we can distinguish: 1) "<4 ms latency required" – indicates a critically important low latency between two nodes; 2) "authentication via" – signifies that one service depends on another for authentication.

Transforming the data into graphs revealed previously undocumented or non-obvious data transfer routes and dependencies. For example, the chain: RFID Scanner → Edge Server → Cloud DB Replica showed that data from the RFID scanner passes through the edge server before reaching the cloud database replica. Such a route might have been missing from the documentation but became visible after analyzing the graph.

We encoded the graphs as sparse adjacency matrices, where rows/columns correspond to node indices and the matrix values represent dependency weights (latency, throughput).
We modeled the dynamic behavior as follows:

$$Q(t) = AX(t) + Bu(t),$$

where:
$Q(t)$ - Service State Vector (e.g., API response time).
$A$ - Adjacency Matrix (device to service connectivity).
$X(t)$ - Device State Vector (CPU, packet loss).
$Bu(t)$ - Control Inputs (Load Balancing).

The empirical validation included over 78 hours of continuous operation on a live hub. For monitoring, we used SNMP traps, API call logs, and NetFlow records.

Thanks to this formalization, system architecture design has transformed from reactive troubleshooting into a predictable process akin to physical modeling. Our approach has ensured a transition from reactive problem-solving to physically predictable design.

### 4.3 Resource Administration Algorithm

In our implementation, the resource management algorithm for a multimodal logistics center functions as a closed-loop control system. We built its core around a multi-criteria optimization mechanism that continuously recalibrates itself using a live stream of telemetry data from the infrastructure's IoT devices. This design introduces self-regulating capabilities directly into the system architecture, allowing us to solve the problem of fragmented, "patchwork" resource allocation through the unified management of a hybrid cloud and edge environment.

We specifically designed the system to adapt to load changes faster than human reaction times, effectively creating an autonomous "nervous system" for the MLC architecture. At its foundation, a virtual resource orchestrator acts as a dynamic control plane. Its multi-criteria optimization core handles conflicting objectives, like minimizing customs transaction latency while also reducing power consumption, by employing constraint programming and Pareto frontier analysis. To make this possible, the system processes metrics from power and performance sensors in real time, transmitted via specialized IIoT protocols (MQTT/CoAP) that we adapted for Russian equipment. Leveraging these protocols and standards, such as OPC UA TSN, enables millisecond-level data processing and deterministic delays (<15 ms). We found this to be absolutely critical for maintaining control stability in the rapidly changing operational environment of a logistics hub.

The constant stream of telemetry data, pulled from loading equipment, video surveillance systems, storage condition sensors, and customs terminals, empowers the algorithm to avoid short-sighted decisions. For instance, the system can balance power constraints at the edge with strict SLAs for customs data processing latency, even during peak loads. When we validated the system under conditions simulating a 200% load spike (using

IETF RFC 2544 methodology), it confirmed the system's elasticity: it successfully redistributed virtual resources between peripheral Kubernetes nodes and Yandex Cloud services in less than 1 second, all while adhering to reliability thresholds per ISO/IEC 25010.

The core of our solution relies on mathematical models, implemented as Mixed-Integer Linear Programs (MILP), that treat the MLC's cloud and periphery as a single computing continuum. This is far more than an abstraction; it is a formal model where heterogeneous resources, from edge microclusters to powerful cloud servers, are represented as a unified pool with shared constraints and an objective function. This foundational approach is what permits dynamic resource borrowing between tiers.

The model operates with aggregated resource vectors $R_{total} = R_{edge} + R_{cloud}$.

$R_{edge}$ describes available resources of edge nodes (CPU, RAM, SSD IOPS, GPU, network bandwidth).

$R_{cloud}$ describes resources available for allocation from cloud providers (e.g., virtual instances of specific classes).

Binary Placement Variables: Binary (integer) variables $x_{\{i,j\}} \in \{0, 1\}$ are introduced,

where:

$i$ is the identifier of a computational task (e.g., container, virtual machine, serverless function).

$j$ is the identifier of a computational node (physical server on the edge or virtual instance in the cloud).

$x_{\{i,j\}} = 1$ means that task $i$ is placed on resource $j$.

The objective function is a weighted sum of costs and SLA violations that we minimize, for example:

Minimize:

$$\Sigma(C_{cloud} * T_{cloud}) + \Sigma(C_{latency} * L_i) + \Sigma(C_{migration} * M_i)$$

where:

$C_{cloud}$ is the cost of using cloud resources per unit time.

$T_{cloud}$ is the cloud usage time.

$C_{latency}$ is the penalty for latency for critical task $i$ (e.g., processing customs transactions).

$L_i$ is the execution latency of task $i$.

$C_{migration}$ is the cost of migrating a task.

$M_i$ is a binary indicator of migration for task $i$.

We realized the "borrowing" mechanism by solving the MILP in real-time based on incoming telemetry. Consider a typical scenario: an edge node processing video analytics from loading equipment exhausts its GPU resources ($R_{edge_{GPU}} \approx 0$), while the task's SLA requires a latency of < 100 ms.

The model's solution process then unfolds as follows:

Latency Constraint: The model is bound by the constraint L_i < 100 ms. Simply placing the task in a remote cloud ($j \in$ Cloud) could violate this constraint due to network latency.

Scenario Evaluation: The model evaluates two primary scenarios:

Option A (Borrowing) involves searching the continuum for another peripheral node with sufficient GPU resources. If the delay is within the limit, the task is migrated.

Option B (Hybrid execution) is used if no free peripheral is available. Here, the model can decide on a hybrid approach. For instance, latency-critical frame preprocessing ($x_{\{preprocess,edge\}} = 1$) remains on the edge, while the resource-intensive neural network inference ($x_{\{inference,cloud_{GPU}\}} = 1$) is offloaded to a powerful GPU instance in the cloud with minimal channel latency (e.g., using dedicated lines).

Optimization Solution: The MILP solver finds a set of values for $x_{\{i,j\}}$ that minimizes the objective function without violating constraints on latency, power consumption ($C_{power}$), and availability.

In another example from our testing, the model can react to telemetry indicating an exhausted power budget on an edge server. This capability allows the scheduler to proactively transfer video analytics tasks from the overloaded edge device to cloud GPUs without interrupting ongoing logistics processes.

The solution we developed fundamentally eliminates the problem of "patchwork" automation by applying unified optimization rules to all components – from cloud VMs to edge sensors in the warehouse. This is particularly vital for MLCs struggling with legacy isolated systems. Our stress testing, which simulated peak loads during customs processing, confirmed the architecture's resilience: the system automatically deployed ephemeral functions in Yandex Cloud Functions, demonstrating a robust capacity to absorb the extreme loads characteristic of multimodal hubs.

### 4.4 Finite Element Model

To make the theoretical model of the system architecture compatible with future digital twins of the multimodal transport hub, we recognized that it had to move beyond traditional static modeling approaches. Our goal was to develop a modeling

framework that could be directly executed for simulation. We achieved this by transforming the conceptual model into executable computational elements using the finite element analysis (FEA) methodology.

For this transformation, we selected Project Studio CS, a Russian software solution known for its proven capability to handle abstract architectural relationships. Much like COMSOL Multiphysics, this tool allows us to convert functional dependencies between physical devices and cloud/edge services into discrete mathematical representations governed by partial differential equations (PDEs). Our process began with a conceptual architectural model created in ArchiCAD, which we then imported into Project Studio CS. There, we transformed it into a finite element mesh complete with specified boundary conditions, loads, and parameters.

To illustrate the model's application, consider one of our key scenarios. The main condition, "Optimization of AI inference task distribution between edge devices and cloud," included an additional focus on "Optimization of edge computing (CPU/NPU + network)." In this setup, edge devices, such as cameras, sensors, smartphones, and industrial controllers, perform AI inference locally, right at the data collection point. The cloud, on the other hand, handles more resource-intensive tasks like model training, global analytics, and long-term storage.

Our mathematical model conceptualizes the system through the following key elements and connections:

a) a "high-speed inter-server data transfer" link between the "computational nodes" and the "cloud layer" nodes.
b) a "hybrid edge-to-cloud connection" link between the "cloud layer" and "peripheral devices (Edge servers)" nodes.

The system is represented as a continuous computing environment $\Omega$, consisting of a cloud domain $\Omega_c$ (GPU cluster) and a peripheral domain $\Omega_e$ (edge devices), interconnected by network interfaces $\Gamma = \Gamma fast \cup \Gamma hybrid$, where:

$\Gamma fast$: NVLink (900 GB/s), InfiniBand (400 GB/s), Ethernet (100 Gb/s);

$\Gamma hybrid$: 5G/LTE/Wi-Fi 6/Fiber Optics/Satellite.

The core idea of the model involves discretizing the system into finite elements, where each element represents a computational node with its associated network connections. Mathematically, this is expressed as $\Omega_c = \bigcup_{e=1}^{N} \Omega_e$, where the domain is divided into finite elements $\Omega_e^i$, each describing an individual computational element $\{nodes_i, connection_{ij}\}$ with its characteristics. Each element is characterized by its computational power $C_i$ (TFLOPS), communication latency $\tau_{ij}$, and bandwidth $B_{ij}$.

The model's objective is to minimize the functional $J(u)$ under the condition of optimal load distribution $u(x,t)$, where $x$ x is the spatial coordinate and $t$ is the data transmission time between $\Omega_c$ and $\Omega_e$. To minimize the functional $J(u)$ with respect to the vector u, it is necessary to solve the system of equations obtained from the stationarity condition $\delta J/\delta u = 0$.

The key equation of the model is the computational load balance equation $K^e u^e = f^e$ for each element, where:

$K^e$ is the analog of the element stiffness matrix in the Finite Element Method (FEM), characterizing the conductivity (interaction intensity) between computational nodes, determined by the latency and bandwidth of network connections;

$u^e$ is the task distribution vector;

$f^e$ is the incoming load vector.

The stiffness matrix takes different forms for different connection types: high-speed connections $K_{det}$ and hybrid connections $K_{hybrid}$.

Data for model validation and boundary conditions were collected and analyzed in the Prometheus monitoring system. The numerical solution of the equation system was implemented in a custom Python solution using NumPy and included the following steps:

a) discretization by partitioning into nodes and connections;
b) local optimization by solving $K^e u^e = f^e$ for each element;
c) global assembly via the construction of $K_{global}$.

The numerical solution we implemented also accounted for network topology adaptation by continuously recalculating hybrid network connections. We configured the model with an element size of 1 computational node and a time step of 10 ms for dynamic balancing. The convergence criterion was a relative change in the functional of <0.1%.

Model verification was a multi-stage process for us. It involved checking the convergence condition, verifying that calculated values matched measured

ones with an error of less than 5%, and ensuring strict compliance with SLAs for latency (<100 ms), jitter (<15 ms), and availability (>99.95%). Furthermore, we calibrated the model according to mandatory hardware specifications (GOST R ISO/IEC 24730-1-2017, GOST R IEC 61511-1-2011, GOST R ISO 8373-2014, GOST R 59385-2021). This crucial step integrated threshold values for energy consumption, temperature conditions, and interface protocols directly into the modeling parameters, ensuring the model reflected real-world limitations.

Finally, we subjected the model to empirical stress testing through cargo simulation at an existing transport hub (TLC Vorsino). Here, we compared simulated loads representing peak logistics scenarios (like the simultaneous handling of rail and road freight) against physical sensor data to quantify tolerances. This end-to-end process effectively transformed our theoretical constructs into a physical simulation engine, where architectural variables such as node density, capacity distribution, and virtualization factors became adjustable parameters in constrained optimization cycles.

The framework we adopted provides three critical capabilities for digital twin compatibility: 1) Predictive what-if analysis for capacity planning, which models scenarios like equipment failure cascades by changing boundary conditions in the PDE solver. 2) Feedback loops for optimization, where outputs like latency at 200% load automatically trigger parameter adjustments using gradient descent algorithms. 3) API-based data exchange between COMSOL solvers and industrial IoT platforms using OPC UA communication standards.

A key advantage of our solution is that it models all components within their actual physical contexts (network, thermal, and vibration for simulating warehouse flows) rather than in abstract workflows. As our validation showed, this approach removes "patchwork" uncertainty, allowing us to pre-validate architectural decisions against both operational standards (GOST) and site-specific empirical data long before physical deployment.

## 5 Conclusion

The conducted research and empirical results allow us to draw four conclusions regarding the viability of the developed model as a professional reference standard for the system architecture of a multimodal transport hub. First, the comprehensive validation confirmed that individual solutions and fragmented (“patchwork”) optimization represent dead-end approaches for the system architecture. Second, the model can be considered a professional good practice due to the inclusion of four proven components. 1) Resource optimization methods that provide scaling of physical devices and routing of control applications, which demonstrated parameterized load balancing strategies that comply with security policies and integration restrictions of Russian hardware and software, while providing import substitution paths for network/hardware equipment. 2) A topology mapping framework for telematics system resource management (physical devices/management applications) applicable to wholesale/distribution centers, logistics coworking spaces, and multimodal hubs that establishes functional dependencies between physical devices and cloud/edge services while providing a conceptualization of the mathematical model. 3) A resource management algorithm that operationalized the interaction of physical device and network management systems through data-driven scheduling decisions for virtual machines, links, and storage based on hardware availability/utilization metrics, operational attributes (power/performance), and real-time parameter monitoring. 4) Results of conceptual mathematical modeling that produced a finite element architecture model to optimize tier (cloud/edge) compute capacity utilization and hardware configuration decisions, with validation proving forward adaptability to multimodal transportation hub digital twins.

*References:*

[1] V. Fialkin and E. Veremeenko, "Characteristics of Traffic Flow Management in Multimodal Transport Hub (by the Example of the Seaport)," *Transportation Research Procedia*, vol. 20, pp. 205-211, 2017, doi: 10.1016/j.trpro.2017.01.053.

[2] Y. Zhou, T. Kundu, M. Goh, and J. Sheu, "Multimodal transportation network centrality analysis for Belt and Road Initiative," *Transportation Research Part E: Logistics and Transportation Review*, vol. 149, p. 102292, 2021, doi: 10.1016/j.tre.2021.102292.

[3] Distributed Computing. Computer Science, Yale University, March 30, 2024, [Online]. https://cpsc.yale.edu/research/primary-areas/distributed-computing (Accessed Date: April 15, 2025).

[4] *(text in Russian)* 20 best practices for building data warehouses in 2024, Fashih Khan, March 20, 2024, [Online].

https://www.astera.com/ru/type/blog/data-warehouse-best-practices/ (Accessed Date: April 15, 2025).

[5] D. Bhattacharya, F. Currim, S. Ram, "Evaluating Distributed Computing Infrastructures: An Empirical Study Comparing Hadoop Deployments on Cloud and Local Systems," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 1075-1088, 1 July-Sept. 2021, doi: 10.1109/TCC.2019.2902377.

[6] K. X. Li, M. Li, Y. Zhu, K. F. Yuen, H. Tong, and H. Zhou, "Smart port: A bibliometric review and future research directions," *Transportation Research Part E: Logistics and Transportation Review*, vol. 174, p. 103098, 2023, doi: 10.1016/j.tre.2023.103098.

[7] E. Palkina, "Transformation of business models of logistics and transportation companies in digital economy," *Transportation Research Procedia*, vol. 63, pp. 2130-2137, 2022, doi: 10.1016/j.trpro.2022.06.239.

[8] S. Pettit, Y. Wang, and A. Beresford, "The impact of digitalization on contemporary and future logistics," in *The Digital Supply Chain*, B. L. MacCarthy and D. Ivanov, Eds. Elsevier, 2022, pp. 111-125, doi: 10.1016/B978-0-323-91614-1.00007-1.

[9] Logistics 4.0 and smart supply chain management in Industry 4.0, *i-SCOOP*, [Online]. https://www.i-scoop.eu/industry-4-0/supply-chain-management-scm-logistics/ (Accessed Date: April 21, 2025).

[10] M. Di Capua, A. Ciaramella, and A. De Prisco, "Machine Learning and Computer Vision for the automation of processes in advanced logistics: the Integrated Logistic Platform (ILP) 4.0," *Procedia Computer Science*, vol. 217, pp. 326-338, 2023, doi: 10.1016/j.procs.2022.12.228.

[11] Y. Lu, K. Morris, S. Frechette, "Current Standards Landscape for Smart Manufacturing Systems," NIST Interagency/Internal Report (NISTIR), *National Institute of Standards and Technology*, Gaithersburg, MD, 2016, doi: 10.6028/NIST.IR.8107.

[12] V. Kubil, V. Mokhov, D. Grinchenkov, "Modelling the generalized multi-objective vehicle routing problem based on costs," *Proceedings of the 6th international conference on applied innovations in IT*. Perm National Research Polytechnic University Anhalt University of Applied Sciences. Koethen. pp. 29-35. 2018, [Online]. https://icaiit.org/proceedings/Proc_of_6th_ICAIIT.pdf (Accessed Date: April 21, 2025).

[13] R. Costa, R. Jardim-Goncalves, P. Figueiras, M. Forcolin, M. Jermol, and R. Stevens, "Smart Cargo for Multimodal Freight Transport: When 'Cloud' becomes 'Fog'," *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 121-126, 2016, doi: 10.1016/j.ifacol.2016.07.561.

[14] B. Lebedev, O. Lebedev, A. Zhiglaty, "Modified Ant Algorithm VLSI Planning on the Basis of the Composite Model of the Solution Space," *Herald of the Bauman Moscow State Technical University. Series Instrument Engineering*. pp. 49-63. 2019.

[15] A. Hakiri, A. Gokhale, S. Ben Yahia, and N. Mellouli, "A comprehensive survey on digital twin for future networks and emerging Internet of Things industry," *Computer Networks*, vol. 244, p. 110350, 2024, doi: 10.1016/j.comnet.2024.110350.

[16] R. Ebera and S. Schwarzera, "Toolbox for Efficient Implementation of Digitalization in Logistics," *Procedia CIRP*, vol. 130, pp. 701-704, 2024, doi: 10.1016/j.procir.2024.10.151.

**Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)**

- Mamedova N.A. – creation of the research model, activities for creating metadata, accumulating research data, preparation and creation of the manuscript draft, obtaining project financial support.
- Afanasev M.A. – conducting the research process, editing the article; Urintsov A.I. – application of formal methods for analysis and synthesis of research data, providing computational resources.
- Haritonov S.V. – data verification, editing the article.

**Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself**

The research was funded by a grant
Russian Science Foundation: № 24-21-20089.

**Conflict of Interest**

The authors have no conflicts of interest to declare.

US