

Article

Petri-Net Modeling and Simulation of a Cloud-IoT-Client Architecture for Education

Liana Luminița Boca ^{1,2,*}, Elisabeta Mihaela Ciortea ³ and Vasile Teodor Dădârlat ^{2,*}

¹ Department of Mathematics and Computer Science, Faculty of Mathematics and Computer Science, Transilvania University of Brasov, 500036 Brasov, Romania

² Computers Department, Faculty of Automation and Computers, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania

³ Department of Computer Science, Mathematics and Electronics, Faculty of Computer Science and Engineering, "1 Decembrie 1918" University of Alba Iulia, 510009 Alba Iulia, Romania; mciorteauab.ro

* Correspondence: liana.boca@unitbv.ro (L.L.B.); vasile.dadarlat@cs.utcluj.ro (V.T.D.)

Abstract

This paper proposes a formally analyzable Cloud-IoT architecture for educational environments, connecting learners, teachers, and institutional administrators to a shared cloud infrastructure through an intermediate IoT layer. The architecture is instantiated in a business-language learning scenario, with four cloud subsystems: a Domain Terminology Repository, an Adaptive Learning Engine, a Monitoring and Assessment Module, and an Administrative and Reporting Module. Its dynamic behavior is formally represented using a timed Petri net incorporating state evolution, transition-delay parameters, feedback, and formal behavioral conditions. The complete model is simulated using Visual Object Net++, and the results are analyzed across the Cloud, IoT, and Client subnets to examine state evolution and marking variation. Additional controlled experiments examine the model under timing variations and increased concurrent Client workloads. A reduced Generalized Stochastic Petri Net (GSPN), representing a Cloud-IoT processing and feedback cycle, is subsequently analyzed using GreatSPN. State-space analysis establishes boundedness, absence of dead markings, and reachability of the response state for the reduced model. Steady-state analysis further shows that reducing an exponentially timed transition rate increases the occupancy of its preceding place. The combined approach provides behavioral evidence for the complete architecture and complementary formal and quantitative results for the reduced model, without extending these results to the complete Petri net.

Keywords: internet of things (IoT); cloud computing; Petri nets; educational technology; formal modeling; simulation; IoT architecture

Academic Editor: Amiya Nayak

Received: 26 August 2026

Revised: 28 September 2026

Accepted: 30 September 2026

Published: 7 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The ongoing advancement of educational technologies has reshaped contemporary learning environments, facilitating real-time data collection, adaptive teaching, and seamless interaction between connected devices, students, and teachers (educators). Within this landscape, the Internet of Things (IoT) presents significant opportunities to enhance both administrative efficiency and pedagogical practice. When combined with cloud computing, IoT enables educational institutions to manage vast amounts of data, provide

tailored learning experiences, and support evidence-based decision-making across classroom and digital learning contexts.

Despite the promise of IoT-enabled education, modern institutions face several challenges in their implementation. Educational systems must accommodate diverse learner profiles, monitor progress continuously, and optimize the use of resources—all within increasingly complex digital infrastructures. Fragmented device ecosystems, varying communication protocols, and the absence of unified standards often limit the scalability and interoperability of IoT solutions. Existing literature highlights that IoT can also enhance personalized learning, monitor learner progress, and support adaptive classroom environments, providing practical tools to improve both student outcomes and administrative efficiency. These constraints highlight the need for systems that are not only technically robust but also adaptable to different pedagogical contexts.

Cloud computing plays a crucial role in addressing these challenges by providing centralized storage, processing power, and analytics capabilities. When combined with IoT devices, cloud platforms can support real-time feedback, personalized learning pathways, and data-driven administrative decisions. They also enable secure data storage, on-demand access to learning resources, and tailored recommendations based on learner behavior, creating a more responsive and connected educational environment. Furthermore, accessible IoT development platforms, such as Arduino and ESP8266-based development boards, provide practical tools for students and teachers to prototype and experiment, supporting hands-on learning and the development of IoT-based educational applications. Together, IoT and cloud technologies offer a foundation for building coherent systems capable of integrating multiple educational functions while supporting informed, adaptive teaching practices.

One of the central considerations of this study is to explore how an integrated Cloud-IoT architecture can support educational processes. In particular, the following questions guide our investigation:

- Can a single Cloud-IoT architecture accommodate multiple categories of educational clients within one coherent framework?
- How can the dynamic behavior of such an integrated system—including timing, feedback mechanisms, and formal behavioral conditions—be represented and analyzed using Petri nets?
- What does simulation of this architecture reveal about state evolution and marking variation across the Cloud, IoT, and Client layers?

Addressing these questions requires moving beyond architectures that are described only at a conceptual level, toward one that is formally analyzable and capable of representing multiple categories of educational clients. To this end, we propose a Cloud-IoT architecture that connects three categories of educational clients to a shared cloud infrastructure through an intermediate IoT layer, and we define its dynamic behavior formally using a timed Petri net. This formalization makes it possible to represent the system's evolution through a global state equation, characterize its feedback mechanism, and define formal conditions concerning conservativeness, deadlock avoidance, and buffer limitation. Latency, throughput, and response rate are additionally identified as relevant measures for subsequent quantitative performance evaluation. The resulting model is then simulated using Visual Object Net++, allowing the behavior of the architecture to be observed under controlled simulated conditions before any physical deployment is considered.

The contribution of this paper is therefore threefold: first, the extension of the Cloud-IoT educational architecture to explicitly represent three categories of educational clients within the formal model; second, a timed Petri-net formalization of this architecture, enabling its dynamic information-flow behavior to be examined through simulation under

controlled timing and workload variations; and third, complementary state-space verification and steady-state analysis of a reduced GSPN representation of a representative Cloud-IoT processing and feedback cycle. This architectural organization is positioned relative to established IoT reference architectures—notably the IoT-A (Internet of Things Architecture) Architectural Reference Model [1] and IEEE Std 2413-2019 [2]—which define the shared architectural viewpoints and concerns that a general-purpose IoT architecture is expected to address; the present architecture specializes these general concerns to the educational client domain rather than proposing a new reference-architecture class. To demonstrate its applicability, the proposed architecture is instantiated in Section 3 through a business-language learning scenario, in which learners, teachers, and institutional administrators are represented as three structurally distinct client categories.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature and identifies the gap motivating this study; Section 3 introduces the proposed architecture; Section 4 formally defines its dynamic behavior using a timed Petri net; Section 5 presents the simulation-based analysis of the complete model, the complementary formal and steady-state analysis of the reduced GSPN, and the resulting architectural interpretation and limitations; and Section 6 concludes the paper.

2. Literature Review

2.1. Review Approach and Search Strategy

To position the proposed architecture within the existing literature, a targeted narrative review was conducted focusing on studies relevant to IoT and cloud-based educational systems and their formal modeling. The review was intended to identify the architectural and modeling approaches most directly related to the present study rather than to provide an exhaustive systematic mapping.

Relevant literature was identified through IEEE Xplore, ACM Digital Library, Web of Science, and Google Scholar. The search focused on IoT and cloud-based architectures for educational environments, formal modeling of IoT and distributed systems, and Petri-net-based approaches, including timed and stochastic models. Priority was given to peer-reviewed studies addressing architectural design, dynamic behavior, formal analysis, or performance-related aspects, together with established reference architectures and foundational works required to support the proposed formal framework.

2.2. Thematic Overview

An examination of existing research is essential to outline the principal directions and themes associated with the use of IoT in education. This need is reinforced by the growing scale of research activity in the field. A recent bibliometric analysis of Scopus-indexed publications from 2016 to 2024 shows that scholarly output on IoT in education followed a general upward trajectory, peaking in 2023 before a slight decline in 2024 [3]. Despite this expansion, the same analysis notes that research on IoT implementation in education remains fragmented, lacking a comprehensive and unified understanding of global trends and their practical contributions to educational advancement [3]. This observation reinforces the rationale for the present review: rather than treating IoT-in-education research as a single, homogeneous body of work, it is more productive to organize it around the principal themes that have emerged to date, and to examine how these themes relate to one another.

Existing literature identifies several overarching themes related to the use of Internet of Things (IoT) technologies and devices in education, reflecting both the opportunities and the growing interest in encouraging student engagement with such technologies [4,5]. These include:

- Security practices for IoT devices, such as network traffic regulation, segmentation, and protection against unauthorized access [6].
- Practical applications of IoT in educational environments, including automated classroom adjustments based on student presence, access control mechanisms, attendance tracking, and staff scheduling systems [4].
- The potential of IoT to support personalized and adaptive learning through connected devices, contextual data, and data analysis [7,8].
- Pedagogical and communication considerations associated with the integration of IoT and smart devices into teaching and learning processes [9].

While these themes are well documented, they capture only part of the broader role that IoT can play in education. Several studies additionally emphasize that the success of IoT initiatives depends on balancing the expectations of institutional management, teachers, and learners while addressing organizational and ethical challenges [4]. In particular, part of this literature treats IoT devices and applications as standalone interventions—addressing security, automation, personalization, or pedagogy individually—without situating them within the broader infrastructure required to support them reliably and at scale. Understanding the cloud computing layer underpinning these applications is therefore essential to clarify both the opportunities and the practical constraints shaping current IoT-enabled educational systems.

This motivates the present study's focus on cloud-based IoT infrastructures and accessible IoT development platforms as complementary components of educational deployment. Cloud-based infrastructures provide the storage, processing, and data-management capabilities required to support information generated by connected devices, while development platforms provide a practical means of prototyping and testing IoT-based educational applications. This distinction is particularly relevant in educational environments, where scalable cloud services and accessible hardware platforms can support different stages of system development, from experimentation to integration within broader Cloud-IoT architectures.

The integration of cloud infrastructures into IoT-enabled educational systems is a recurring theme in contemporary research. Numerous studies emphasize that cloud platforms constitute a foundational component for processing, storing, and analyzing data generated by IoT devices across local, hybrid, and fully cloud-based environments. These platforms also enable advanced analytics, including machine-learning-driven insights that support adaptive learning and institutional decision-making. For instance, Botta et al. [10] highlight that the integration of cloud computing with the Internet of Things enables scalable resource provisioning, efficient data storage, and advanced data processing capabilities, which are essential for handling the large volumes of heterogeneous data generated by IoT environments [10].

Within educational contexts, IoT systems can support personalized and adaptive learning by using data from connected devices and learning environments to tailor learning content and pace to individual needs and preferences [8]. More broadly, Al-Emran et al. [11] identify a range of opportunities for IoT adoption across different educational contexts, while also highlighting challenges associated with its integration [11].

Notwithstanding these contributions, many architecture-focused studies in educational IoT remain descriptive, application-specific, or implementation-oriented, without providing an explicit formal model for analyzing the dynamic behavior of the complete architecture. Qureshi et al. [6], for instance, propose a dedicated IoT-based system for school security and monitoring, illustrating the value of targeted applications but also the tendency toward narrowly scoped solutions [6]. At a broader level, Prasetya, Rofiudin, and Herwanto [7] identify persistent implementation challenges, including

infrastructural limitations, data privacy concerns, and limited digital readiness among teaching staff [7].

Similarly, de Lima et al. [12] found that most of the reviewed research on IoT in education focused on technical and utilitarian aspects, while pedagogically oriented theoretical and methodological approaches remained less prominent [12]. Taken together, these studies indicate a more specific gap at the intersection of educational IoT architecture and formal system modeling. Existing work provides educational IoT architectures, application-specific implementations, and formal or stochastic modeling approaches for IoT and distributed systems, but the reviewed literature does not address the specific combination of a multi-stakeholder educational Cloud-IoT architecture, comprising learner, teacher, and administrator roles, within a single executable Petri-net model. This gap motivates the present study, which extends a previously proposed Cloud-IoT architecture by introducing a timed Petri-net representation, examining its cross-layer information-flow behavior through simulation, and complementing this analysis with formal state-space and steady-state evaluation of a reduced GSPN model.

To examine this gap in greater detail, a closer look at architecture-level implementations reveals how the individual pillars of security, personalization, and pedagogy have been combined in practice—and where current designs still fall short of the integrated, formally analyzable model proposed in this paper. Table 1 summarizes sixteen representative architecture-focused studies identified through this review.

Table 1. Representative IoT-cloud architectures and applications in educational settings.

Study	Level	Technological Focus	Contribution
Jurva et al. (2020) [13]	Campus	5G, IoT, edge computing/cloud	Technical architecture and an operational governance model addressing who runs a growing campus IoT infrastructure
Nie (2013) [14]	Campus	IoT + cloud computing	One of the earliest three-layer smart-campus models combining an IoT sensing framework with an education cloud platform
Fraga-Lamas et al. (2019) [15]	Campus	LoRaWAN + fog computing	Empirically validated LPWAN/fog architecture extending cloud computing to enable scalable, low-latency, and energy-efficient smart campus IoT
Pexyeen et al. (2022) [16]	Campus	IoT + AI + digital twin	Smart campus approach integrating IoT, AI, and Digital Twin technologies
Haggag et al. (2025) [17]	Campus	Survey—IoT, AI, sustainability	Comprehensive review of AI, IoT, and Digital Twin technologies for sustainable smart campuses
Koukaras et al. (2025) [18]	Campus	AI + wireless networks	Conceptual framework integrating AI with 5G/IoT infrastructure for smart campus resource allocation and administrative decision-making.
Elbertsen et al. (2025) [19]	Campus	AI + IoT + user experience	User-centered smart campus design framework derived from expert interviews and user validation, identifying key design elements that enhance user experience.

Arsenijević et al. (2026) [20]	Campus	IoT readiness assessment	Context-specific IoT readiness assessment framework (24 dimensions) integrating organizational, technological, and client readiness, evaluated through a university case study.
Burunkaya & Duraklar (2022) [21]	Classroom	IoT sensors + control of environmental factors	IoT-based smart classroom incubator for automated control of classroom environmental conditions tailored to individual needs.
Dai et al. (2023) [22]	Classroom	IoT + Cloud-edge architecture	Four-layer cloud-edge computing architecture distributing data processing across edge nodes for real-time smart classroom applications.
Sharma & Aarthya (2016) [23]	Classroom	IoT + RFID + cloud	Early practical cloud-connected IoT approach for automated attendance management.
Dhir et al. (2024) [24]	Classroom	IoT + pedagogy	IoT in education framework emphasizes enhanced learning experiences, smart classroom applications, and educational process automation.
Jayousi et al. (2025) [25]	Classroom	IoT + inclusive pedagogy	Conceptual framework integrating IoT-based real-time monitoring with adaptive teaching for inclusive education.
Quevedo Piratova et al. (2025) [26]	Classroom	AI + IoT-enabled classrooms	Systematic review of 90 studies identifying architectural, environmental, and technological design principles for AI- and IoT-enabled classrooms
Khashan et al. (2023) [27]	Cross-cutting	IoT + Blockchain authentication	Blockchain-based, energy-efficient decentralized authentication for resource-constrained IoT devices across a multi-university network.
Andriulo et al. (2024) [28]	Cross-cutting	Edge + cloud + hybrid IoT computing	Systematic review comparing edge, cloud, and hybrid architectures for latency-, privacy-, and scalability-sensitive IoT applications.

Read together, these sixteen studies, spanning research published between 2013 and 2026, reveal a clear architectural progression driven by three converging concerns. First, scalability: early designs such as Nie's [14] proposed a centralized, cloud-based three-layer model for campus-wide IoT—an approach that later studies increasingly complemented with edge and fog layers as device counts and latency-sensitive applications grew, favoring computation closer to the data source. This evolution is illustrated by Fraga-Lamas et al. [15], who propose a fog-computing architecture over LoRaWAN specifically to overcome the scalability and power-consumption limits of pure cloud designs, while Dai et al. [22] propose a comparable cloud-edge hybrid for classroom environments to reduce latency and network congestion. Andriulo et al.'s review [28] frames this as a general trade-off in the field: cloud computing offers scalability and flexibility, while edge and fog layers reduce latency and improve privacy, so the right architecture depends on the application requirements and deployment constraints. More recent surveys build on this same infrastructure from different angles: Haggag et al. [17] document a broad set of sustainability-oriented technologies now layered onto smart-campus infrastructure, Koukaras et al. [18] propose extending it toward AI-driven resource allocation and

administrative decision-making, and Arsenijević et al. [20] show, through a university case study, that organizational and cultural factors can become the primary barriers once a campus attempts to implement such an architecture.

Second, purpose: narrowly scoped deployments—environmental control [21], attendance tracking [23]—confirm the technical feasibility of cloud-connected IoT in ordinary classrooms. Dhir et al. [24] broaden this view, discussing the use of IoT-enabled smart classes to support teaching and learning, though their account remains largely exploratory. Two recent studies narrow the focus further: Jayousi et al. [25] propose a conceptual, not yet validated, framework using IoT-based monitoring to support inclusive teaching, while Quevedo Piratova et al.'s review of ninety studies [26] identifies the physical design features—acoustics, lighting, flexible furniture, sensory-inclusive layouts—that define effective AI/IoT classrooms. At the campus level, Elbertsen et al. [19] identify user priorities such as workplace availability and energy efficiency through expert interviews, while Pexyea et al. [16] extend the infrastructure toward AI- and digital-twin-based predictive management. Third, largely unaddressed elsewhere, is security: Khashan et al. [27] identify limitations associated with centralized authentication and propose a decentralized, blockchain-based alternative for resource-constrained educational hardware.

Taken together, the studies summarized in Table 1 primarily address architectural design, implementation, scalability, security, or application-specific concerns rather than the formal representation of the complete educational IoT architecture's dynamic behavior. This distinction motivates the following examination of formal modeling approaches capable of representing concurrency, synchronization, temporal behavior, and uncertainty in distributed systems.

Among these approaches, Petri nets represent one of the most established formalisms for analyzing the dynamic behavior of complex distributed systems. Petri nets are particularly suitable for the architecture considered in this study because the Cloud, IoT, and Client components operate as interconnected processes whose activities may occur concurrently and require synchronization across architectural levels. Their state-based representation also makes it possible to follow the movement of information through the system, while timed transitions represent delays associated with different processing stages. In addition, the feedback path can be represented explicitly within the same formal model. These characteristics make Petri nets appropriate for examining the concurrent, temporal, and iterative behavior considered in the proposed Cloud-IoT-Client architecture. To identify appropriate formal modeling approaches for educational IoT systems, the reviewed studies were selected based on three criteria: (i) the use of formal methods for analyzing system behavior, (ii) their relevance to IoT, educational, or distributed computing environments, and (iii) their capability to address dynamic properties such as concurrency, synchronization, timing, or uncertainty. Table 2 summarizes representative studies according to these criteria and highlights the remaining modeling limitations.

Building on prior work by the authors [29], which proposed and simulated a Petri-net-based IoT architecture for higher education using functional blocks, the present study extends that foundation in three specific ways: (i) it formalizes the architecture as a timed Petri net with an explicit global state equation, feedback relation, and formally stated conditions for conservativeness, deadlock avoidance, and buffer limitation, none of which were defined in [29]; (ii) it complements the simulation-based analysis with formal state-space verification of a reduced GSPN using GreatSPN, establishing boundedness, absence of dead markings, and reachability of the response state within the reduced model; and (iii) it provides a quantitative steady-state analysis of transition-rate variations for the reduced GSPN, an analytical perspective not considered in [29]. Together, these additions

substantially extend the formal and analytical treatment of the architecture beyond the earlier study.

Table 2. Positioning of the present study relative to existing approaches.

Study	Domain	Formal Notation/ Modeling Approach	Analytical Focus	Identified Contribution and Remaining Limitation
Murata (1989) [30]	General formal modeling	Classical Petri nets	Reachability, liveness, boundedness, invariants, deadlock-related properties	Establishes the theoretical foundation of Petri net analysis but does not address time-dependent or stochastic system behavior.
Ahmad et al. (2023) [31]	IoT systems— network security	Colored Petri nets	Security properties, protocol behavior, routing interactions	Demonstrates formal analysis in IoT networks but focuses on security rather than educational architecture or performance behavior.
Fortas et al. (2025) [32]	IoT systems— architecture modeling	High-level Petri nets (G-Nets)	Component interaction, modular behavior, formal verification	Supports formal IoT system modeling but does not analyze temporal constraints or operational performance.
Balogh et al. (2013) [33]	Educational systems—e- learning process modeling	Petri nets	Learning workflows, student progression, process execution	Demonstrates the applicability of Petri nets in education but excludes IoT infrastructure and distributed system dynamics.
Balogh & Kuchárik (2019) [34]	Educational systems – LMS analytics	Petri nets	Correlational/predictive modeling of student grades from LMS usage	Extends the earlier deterministic model (same research group—[33]) toward predictive analysis; still excludes IoT infrastructure and distributed system dynamics.
Khamparia and Pandey (2016) [35]	Educational systems—security modeling	Aspect-oriented Stochastic Petri nets	Security threats, reliability, uncertainty associated with attacks	Applies stochastic Petri nets to threat modeling and security assessment in an e-learning system; does not address a Cloud-IoT educational architecture or cross-layer information-flow analysis.
Boca et al. (2023) [29]	Educational IoT systems— higher education	Petri-net-based modeling and simulation in Visual Object Net++	IoT architecture, information flow, and simulation of system behavior	Proposes and simulates a Petri-net-based IoT architecture for higher education; does not provide the explicit formal property analysis or reduced-model state-space and steady-state evaluation developed in the present study.
Yang et al. (2024) [36]	IoT services	High-level Petri nets	Conflict detection, concurrent interactions	Formal conflict detection in IoT service composition; no educational architecture or performance evaluation.
Hammami & Mathkour (2015)	Educational systems –	Object Petri nets (agent- based)	Agent synchronization, adaptive behavior	Applies Petri nets to adaptive e-learning agents; no Cloud-IoT

[37]	adaptive e-learning			infrastructure or stochastic evaluation.
Vardakis et al. (2025) [38]	Edge-cloud continuum	Petri nets	Resource-scheduling policies, multi-cluster simulation	Models scheduling in edge-cloud continuum; not applied to educational client architecture.
Sanahmadi et al. (2023) [39]	IoT energy consumption	Stochastic Reward Nets (SRN)	Quantitative energy-consumption evaluation	Quantitative stochastic evaluation of IoT energy consumption; not instantiated for an educational architecture.
Melo et al. (2026) [40]	IoT dependability (Lo-RaMesh)	Stochastic Petri nets	Dependability, sensitivity analysis	Stochastic modeling + sensitivity analysis; single-node IoT, not a multi-stakeholder educational system.
Gu et al. (2024) [41]	General Petri net theory	Bounded PN, basis reachability graphs	Liveness/deadlock-freeness verification	Formal verification method; not applied to IoT or educational systems.
Andrade et al. (2021) [42]	Fog-cloud IoT infrastructure	Deterministic and Stochastic Petri Nets (DSPN)	Utilization, response time, throughput, availability	Establishes quantitative performance/availability evaluation for fog-cloud IoT infrastructure generally but is not instantiated for a multi-stakeholder educational client architecture.
Nguyen et al. (2021) [43]	Edge/fog computing—medical information systems	Stochastic Reward Nets (SRN)	Performability metrics (response time, throughput, drop probability, queue utilization) under load-balancing and fail-over strategies	Provides comprehensive quantitative performability evaluation for edge/fog infrastructure using SRN; application domain is healthcare, not education.
Present study	Cloud-IoT-Client architecture for higher education	Timed Petri Net (TPN) with complementary reduced GSPN	Formal modeling of system states, interactions, feedback, and timing; state-space and steady-state analysis of a reduced GSPN	Executable formal model of a multi-stakeholder educational Cloud-IoT architecture; complementary state-space verification and steady-state analysis of a reduced GSPN model.

The comparison presented in Table 2 shows that existing formal approaches address specific aspects of IoT systems, educational processes, security, or distributed infrastructure. Hammami and Mathkour [37] use Object Petri nets to represent communication and synchronization among agents in an adaptive e-learning environment, but their model does not incorporate a Cloud-IoT infrastructure. In the IoT domain, Fortas et al. [32] employ high-level Petri nets for component interaction and formal verification, without focusing on educational stakeholders or temporal performance. Andrade et al. [42], in turn, use deterministic and stochastic Petri nets to evaluate response time, throughput, utilization, and availability in fog–cloud IoT infrastructure, but outside an educational setting.

Stochastic Petri-net methods combining temporal and stochastic behavior with quantitative performance or dependability evaluation have been applied to cloud-fog-IoT and related IoT infrastructures [39,40,42,43]. However, the reviewed studies do not address the specific combination of a multi-stakeholder educational client architecture, comprising learner, teacher, and administrator roles, within a single formal Cloud-IoT model. This application-oriented gap motivates the present study. A related but distinct line of work is educational process mining, which discovers process models—frequently expressed as Petri nets—directly from educational event logs, rather than specifying them a priori [44].

This empirical, data-driven approach complements the analytical formalization pursued in the present study; workflow-net formalisms and their associated correctness properties provide another established use of Petri-net analysis for process-oriented systems [45].

Accordingly, this study extends the previously proposed Cloud-IoT architecture for higher education [29] by introducing a timed Petri-net representation of its dynamic behavior. The architectural concept itself is therefore not presented here as entirely new; the main contribution of the present study lies in its further formalization and analysis. While the previous work used functional blocks together with Petri-net simulation to represent and examine the proposed educational IoT system, the present study extends this foundation through an explicit timed Petri-net formalization of system states, inter-layer interactions, feedback mechanisms, and formal conditions concerning conservativeness, deadlock avoidance, and buffer limitation. The resulting model is implemented and simulated to examine the information-flow behavior of the proposed architecture under the selected model configuration. This formalization provides an executable basis for examining the system's dynamic and information-flow behavior. Complementary formal verification and steady-state analysis are subsequently performed on a reduced GSPN representation, while exhaustive formal verification and quantitative performance evaluation of the complete model remain outside the scope of the present study.

The remainder of this section presents the proposed Cloud-IoT architecture and its main functional components, which provide the foundation for the formal model developed in this study.

3. Proposed Cloud-IoT Architecture

This section presents the proposed cloud-based architecture for educational IoT environments and describes its functional organization. The architecture is designed to provide the infrastructure foundation for the formal dynamic analysis introduced in this study, where cloud services, IoT devices, and communication mechanisms are represented through a timed Petri-net model. The cloud layer serves as a key component for data storage, processing, and analysis, supporting different deployment scenarios, including local, hybrid, and fully cloud-based configurations [28]. Furthermore, the integration of machine learning-based analytics across edge and cloud infrastructures has been explored to enhance the performance and decision-making capabilities of smart education systems [46]. In order to support the formal analysis developed in Section 4, the architectural components introduced below are explicitly labeled as places and transitions of the underlying Petri net.

Within educational environments, the integration of cloud services with IoT technologies offers a wide range of applications and benefits, including [4]:

- Data storage and management: Information gathered by IoT devices can be stored and processed within cloud and Big Data infrastructures supporting educational applications [47].
- Personalized learning: Data on learner behavior, progress, and performance—collected through IoT devices—can be processed in the cloud to support personalized learning experiences. A smart education system can further analyze this data to generate tailored recommendations or monitoring activities [8].
- Monitoring and feedback systems: IoT, cloud, and fog technologies can support monitoring, feedback, and information exchange among students, teachers, parents, and administrators in smart educational environments [48].
- Enhanced access to resources: Smart devices and networked learning environments facilitate access to digital learning resources and support personalized and seamless learning [49].

- **Administrative efficiency:** A connected IoT infrastructure enables the collection, organization, and real-time analysis of administrative data, supporting informed decision-making and streamlined institutional management [50].
- **Communication and collaboration:** IoT technologies can support collaboration among learners and instructors by enabling connected devices and applications to exchange information across the learning environment [51].

To ground the proposed architecture in a concrete pedagogical context, the three client categories are instantiated within a business-language learning environment, corresponding to three stakeholder roles relevant to IoT-enabled educational environments [4]: learners, teachers, and institutional administrators. Client 1 represents the learner, who engages with business-language content and receives personalized learning recommendations. Client 2 represents the teacher, who monitors learner progress, adjusts instructional content, and provides feedback. Client 3 represents the administrator, who is responsible for institutional oversight, resource allocation, and decision-making based on aggregated system data. Although situated within a single specialized-language domain, these three roles have distinct objectives and interact with different subsystems of the architecture, supporting their representation as structurally distinct client categories. Correspondingly, the cloud layer comprises four core processing subsystems: a Domain Terminology Repository, an Adaptive Learning Engine, a Monitoring and Assessment Module, and an Administrative and Reporting Module. The latter supports the decision-making function described in the Abstract by providing the administrator with consolidated information on learner progress, system activity, and assessment outcomes.

The continuous collection of learner behavioral and performance data by the Monitoring and Assessment Module, and its transmission to the Administrative and Reporting Module for institutional oversight and resource allocation, constitutes personal-data processing within the meaning of EU data protection law [52]. This raises data minimization and purpose limitation as necessary design constraints for any practical deployment of the proposed architecture [52], together with broader privacy and ethical considerations associated with learning analytics [53,54], particularly given that data privacy has already been identified as a barrier to IoT adoption in education more broadly [7].

The cloud layer represents the central information and processing environment of the proposed architecture. It stores and manages the main resources required for business-language learning, including domain-specific data, learning materials, texts, terminology, exercises, tests, assessment results, learner profiles, and personalized learning plans. The cloud therefore provides a shared repository through which the three client roles can access, update, and exchange the information relevant to their respective objectives.

IoT Cloud refers to the cloud-based infrastructure layer that supports IoT-generated data throughout its lifecycle. Rather than referring to a specific platform, it encompasses the cloud services that typically support IoT applications, including data storage, computation, and analytics. By connecting IoT devices to these centralized services, IoT Cloud enables real-time synchronization, scalable data processing, and remote access across distributed educational environments.

Open-source IoT development platforms, including microcontroller-based platforms such as NodeMCU ESP8266, can be used to translate IoT concepts into demonstration-level prototypes. Their accessibility allows students and educators to design, test, and evaluate IoT-based solutions in educational settings. Such platforms are particularly useful for prototyping and experimentation before integration into larger Cloud-IoT architectures.

Despite persistent interoperability constraints caused by the lack of unified standards across IoT layers, programming languages, and vendor-specific communication protocols, open platforms continue to play a pivotal role in expanding access to IoT ecosystems.

They lower development barriers and enable rapid prototyping and experimentation across diverse application domains. Standardization efforts such as ITU-T Y.2060 [55] and IEEE Std 2413-2019 [2] address IoT architectural organization from complementary perspectives. ITU-T Y.2060 defines a four-layer IoT reference model, while IEEE Std 2413-2019 provides an architectural framework based on shared stakeholder concerns and architecture viewpoints. The three-layer Cloud-IoT-Client organization adopted in Section 3 can be viewed as a domain-specific specialization of the ITU-T model, with the application and service-support layers represented within the Cloud subsystem. In the educational sector, [56] provide a structured survey of IoT deployment in tertiary institutions, highlighting its integration across multiple functional modules and its contribution to emerging smart learning environments. Their review identifies a broad range of IoT applications and potential benefits in tertiary education, while also emphasizing the implementation challenges that can limit their adoption [56]. However, the study also emphasizes that the realization of these benefits is strongly dependent on overcoming infrastructural limitations and addressing implementation challenges, particularly in resource-constrained educational contexts [56].

Figure 1 illustrates the general architecture proposed for the system. All foundational information is stored in the cloud, where it is organized into four dedicated subsystems. IoT technologies enable real-time communication between the clients and the cloud through their available devices. The information processed within this architecture is specific to the business-language learning domain and is accessed differently by the three client roles according to their respective objectives.

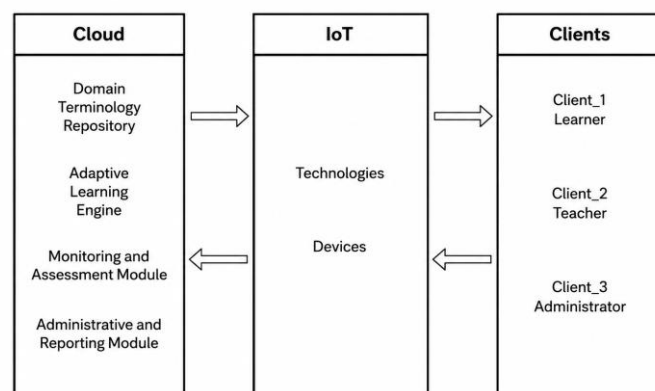


Figure 1. General architecture.

The architectural components described above correspond conceptually to the three interconnected subnets of the Petri net model introduced in Section 4 and simulated in Section 5 (Figure 2). The cloud subsystems (Domain Terminology Repository, Adaptive Learning Engine, Monitoring and Assessment Module, and Administrative and Reporting Module) are represented within the *Cloud* subnet; the IoT technological and device layers are represented within the *IoT* subnet; and the three client roles (learner, teacher, and administrator) are represented by the *Client_1*, *Client_2*, and *Client_3* subnets, respectively. This correspondence is established at the level of architectural subsystems rather than individual places and transitions, as the complete formal model comprises a considerably larger number of places and transitions, as shown in Figure 2.

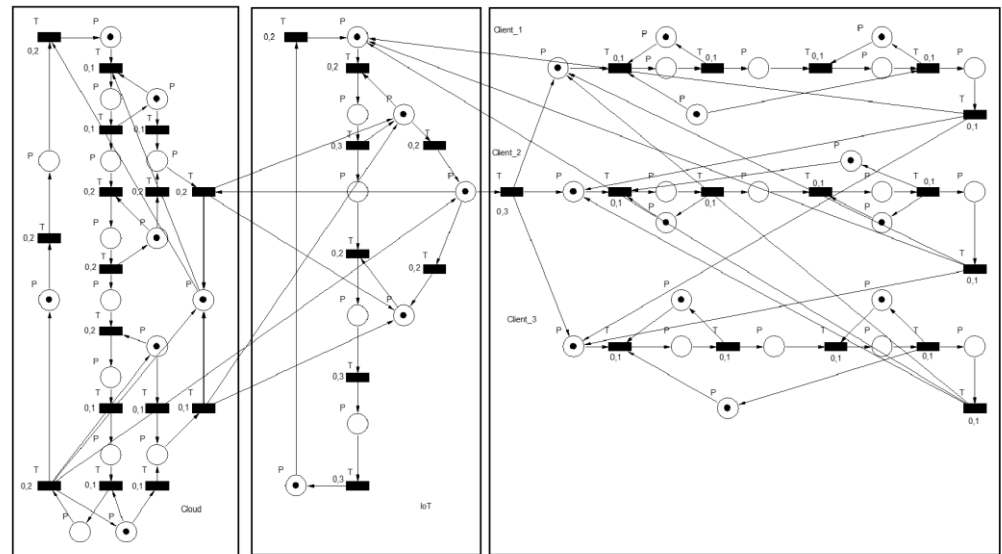


Figure 2. Timed Petri-net model of the Cloud-IoT-Client architecture, comprising three interconnected subnets—Cloud, IoT, and Client (instantiated as three client roles, Client_1–Client_3)—with places, transitions, per-transition delay values, and the initial marking.

4. Formal Model (Timed Petri Net)

The formal foundation of this model follows the classical Petri net framework [30], which provides the structural and behavioral properties required for a dynamic and formally analyzable representation of such systems. Timed and stochastic extensions of Petri nets [57–59] provide the broader methodological context for the temporal representation used in the complete model and the reduced GSPN analysis introduced subsequently in Section 5.7. This approach is further motivated by applications of Petri-net-based modeling to IoT and educational systems [31–35], as well as by the increasing complexity of contemporary IoT and cloud-based deployments [60,61]. The state of the system at each subsequent time instant results from the cumulative effects of the processes occurring within it, and its dynamic evolution can be formally described by a global state equation.

The system is defined as follows (1):

$$\mathcal{N} = (P, T, F, W, M_0, \Theta) \quad (1)$$

where

- $P = \{p_1, p_2, \dots, p_n\}$ —the set of places (circles)
- $T = \{t_1, t_2, \dots, t_m\}$ —the set of transitions (rectangles)
- $F \subseteq (P \times T) \cup (T \times P)$ —the set of arcs
- W —the weight function
- M_0 —the initial marking
- Θ —the vector of transition-delay parameters associated with the timed transitions of the model (0.1, 0.2, and 0.3 model time units).

In the complete model, the documented numerical temporal parameters are the transition-delay values shown in Figure 2. No explicit probability distributions or stochastic transition rates are assigned to this model; stochastic behavior is examined quantitatively only in the reduced GSPN introduced in Section 5.7.

In the following, several mathematical elements are introduced in order to define a global model, as follows:

- Global state equation [30]:

$$M_{total}(k+1) = M_{total}(k) + C_{total} \cdot \sigma(k). \quad (2)$$

Equation (2) not only represents a mathematical relation, but also conceptually expresses that each system state is the cumulative result of all preceding transformations. Each active process—whether local processing at the IoT level, complex analysis in the cloud, or an action executed at the client level—contributes to modifying the distribution of information within the system. Thus, the system does not evolve linearly, but through a mechanism of continuous accumulation and redistribution of informational resources.

For the proposed architecture, the feedback mechanism is represented as:

- Feedback loop:

$$M_{in}(k+1) = f(M_{out}(k)) \quad (3)$$

where $M_{out}(k)$ denotes the output marking resulting from the processing performed at step k , $M_{in}(k+1)$ denotes the input marking for the subsequent processing step, and f represents the feedback mapping through which processed information is returned to the system.

The feedback relation in Equation (3) represents the return of processed information from the Cloud level to the operational components of the architecture. The resulting actions may generate new data that re-enter the processing cycle, thereby representing the iterative information flow between the Client, IoT, and Cloud levels.

- Performance measures:

- L —Latency
- T —Throughput
- R —Response rate

The performance of the system can be characterized by using latency, throughput, and response rate. Latency denotes the time required for information to traverse the system and for a response to be produced, throughput denotes the amount of information processed per unit of time, and response rate describes the rate at which the system produces responses to incoming events. These measures identify relevant dimensions for a subsequent quantitative performance evaluation of the model. They are not numerically evaluated in the present simulation.

Several formal properties are relevant when examining the admissible behavior of the proposed model. The first property considered here is conservativeness, which concerns the preservation of a defined token quantity across the evolution of the Petri net [30].

- Conservativeness [30]:

$$\sum_i M(p_i) = \text{const} \quad (4)$$

a special case of the general invariant condition with unit place weights.

A second relevant property is deadlock avoidance. In Petri-net terms, a dead marking is a reachable marking from which no transition can fire. Examining such states is therefore relevant to determining whether execution can continue from the reachable markings of the model.

- Deadlock avoidance [30]:

$$\forall M \in R(M_0), \exists t: M[t] \quad (5)$$

- Buffer limitation [30]:

$$M(p_i) \leq M_{max} \quad (6)$$

Buffer limitation concerns the restriction of token accumulation in the places of the model. In the proposed formalization, this property provides a criterion for examining whether the number of tokens associated with a place remains within a defined bound.

Taken together, these elements represent the architecture as a dynamic system in which state evolution results from the interaction of concurrent processes, temporal parameters, and feedback.

Equations (1)–(6) define the structural, behavioral, and temporal elements of the proposed architecture, together with formal conditions concerning conservativeness, deadlock avoidance, and buffer limitation. This formalism is instantiated for the three-subnet structure introduced in Section 3, corresponding to the Cloud, IoT, and Client subsystems, and is subsequently translated into an executable simulation model. Section 5 presents this simulation, implemented in Visual Object Net++, and examines the resulting information flow across the architecture's components.

5. Simulation and Analysis

5.1. Simulation Model and Verification Framework

The formal model introduced in Section 4 (Equation (1)) defines the structural and behavioral properties of the proposed architecture but does not, by itself, show how the system evolves under simulated operating conditions. To operationalize this model, the Petri net is translated into an executable simulation implemented in Visual Object Net++. This step makes it possible to observe the dynamic evolution of the architecture, represented abstractly by the global state equation (Equation (2)), prior to physical deployment.

Visual Object Net++ was selected as the simulation platform because it supports timed Petri nets and allows the parameters of the model to be specified and varied during simulation. The simulated model preserves the three-subnet organization established in Section 3: a Cloud subnet, corresponding to the four core processing subsystems—the Domain Terminology Repository, Adaptive Learning Engine, Monitoring and Assessment Module, and Administrative and Reporting Module—an IoT subnet representing the intermediate orchestration and processing layer, and three Client subnets (Client_1, Client_2, and Client_3), corresponding to the learner, teacher, and administrator roles, respectively. As established in Section 3, this correspondence is maintained at the level of architectural subsystems; consequently, the complete Petri net contains a considerably larger number of places and transitions than the conceptual architecture shown in Figure 1. While Visual Object Net++ supports timed Petri-net simulation, it does not provide state-space analysis capabilities such as invariant computation, reachability graph construction, or liveness verification. These limitations are addressed in two complementary ways in the present study: the complete model is examined through simulation-based behavioral analysis in Sections 5.2–5.6, while a reduced GSPN representation of a representative Cloud-IoT processing and feedback cycle is subjected to formal state-space and steady-state analysis using GreatSPN in Section 5.7. The reduced model is introduced specifically as a complementary verification model and should not be interpreted as a reconstruction or exhaustive verification of the complete Petri net shown in Figure 2.

The models used in the two software environments are provided as Supplementary Materials. The complete model was simulated using Visual Object Net++ version 2.7a, while the reduced GSPN was analyzed using GreatSPN 3 version 1.6.91. The supplementary files are organized by software environment, with an accompanying README file describing the provided models and their use in the analyses presented in this paper.

The complete model comprises three interconnected subnets (Cloud, IoT, and three Client instances), each containing multiple places and transitions with associated firing-delay parameters (0.1, 0.2, or 0.3 model time units, as shown in Figure 2). The structural

configuration available for the present analysis is documented in Figure 2, including the subnet organization, transition-delay parameters, and initial marking.

The simulation incorporates the transition-delay parameters represented in Figure 2, which determine the temporal behavior of transitions across Cloud, IoT, and Client subnets. The delay values used in the complete model are controlled simulation parameters rather than values calibrated from measurements of a deployed educational IoT system. They were used to represent different relative timing conditions within the model and to support the examination of its dynamic behavior under a consistent simulation configuration. Accordingly, the numerical values should be interpreted within the model time scale rather than as estimates of real processing or communication delays. Their role in the present study is therefore comparative: they allow the response of the model to different timing conditions to be examined without assigning the simulated values to a specific deployed system. Published experimental studies of educational IoT environments provide useful reference points for temporal performance. López Camacho et al. [62] report response-time measurements for an implemented IoT-based learning platform under increasing numbers of active devices, while [63] reports latency and data-rate measurements from an educational edge-computing environment under different numbers of concurrent users. However, these measurements are specific to the respective implementations and do not map directly to the individual transitions of the Petri-net model considered here; therefore, they were not used for direct parameter calibration. The resulting marking evolution provides the basis for examining information flow across the interconnected subnets.

The analysis is conducted at four complementary levels. First, the observed behavior of the complete model is examined through the simulated information flows of the Cloud, IoT, and Client subnets (Sections 5.2–5.4). Second, Section 5.5 interprets these observations in relation to the formal model and the architectural organization established in Sections 3 and 4. Third, Section 5.6 extends the complete-model analysis through marking-variation assessment, controlled timing and workload experiments, and discussion of the formal properties introduced in Section 4. Finally, a reduced GSPN representation of a representative Cloud-IoT processing and feedback cycle is analyzed independently in GreatSPN to complement the simulation results with state-space verification and steady-state analysis (Section 5.7). This combined approach distinguishes between behavioral and quantitative evidence obtained from the complete simulated model and formal results established for the reduced verification model.

5.2. Cloud Subsystem Simulation

The Cloud subnet represents the central processing component of the proposed architecture and corresponds to the four core subsystems defined in Section 3: the Domain Terminology Repository, Adaptive Learning Engine, Monitoring and Assessment Module, and Administrative and Reporting Module. In the simulation model, these functional components are represented within the Cloud subnet through a more detailed network of places and transitions than the conceptual representation shown in Figure 1. This distinction reflects the different levels of abstraction between the architectural model and its formal Petri net representation.

Figure 3 presents the marking evolution of three selected places of the Cloud subnet (m_0 , m_{10} , and m_{16}) recorded during a single 5 s baseline simulation run of the complete model shown in Figure 2. The horizontal axis represents simulation time, while the vertical axis represents the marking of each monitored place.

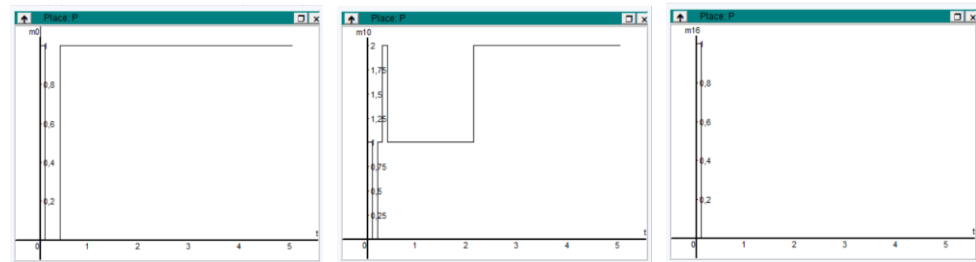


Figure 3. Marking evolution at selected places of the Cloud subnet: m0, m10, and m16.

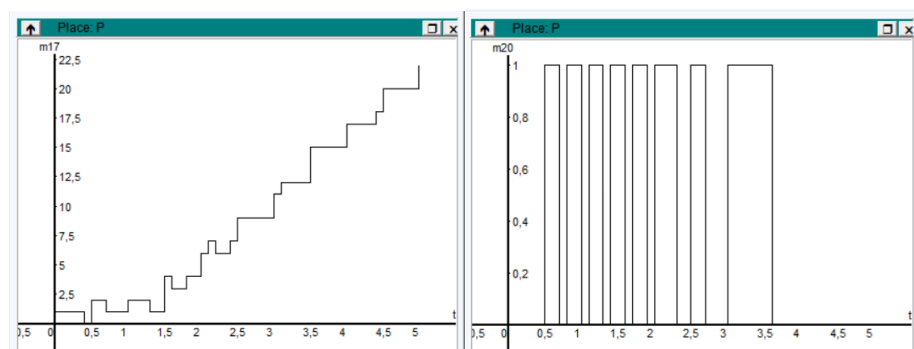
Observation. During this simulation run, the three monitored places exhibit different marking patterns. After the initial transition activity, the markings of m0 and m16 remain at 1 for the rest of the simulated interval. Place m10 follows a different pattern: its marking initially remains at 1 and later increases to 2, where it remains until the end of the run. Thus, within this run, the marking evolution is not uniform across the three monitored Cloud places.

Interpretation. The different marking patterns observed at m0, m10, and m16 show that the selected Cloud places follow different state evolutions during this run. Within the Petri-net model, these changes reflect the firing of transitions and the structural connections between places and transitions. This behavior is consistent with the global state equation (Equation (2)), in which the system marking evolves as a result of transition firings.

Implication. The observed marking evolution provides a direct view of the simulated state changes within selected places of the Cloud subnet under the baseline timing configuration. These results characterize the behavior observed in the simulated Petri-net model and should not be interpreted as quantitative performance measurements of a deployed Cloud system. The distinction between behavioral analysis and quantitative performance evaluation is discussed in Section 5.8. The behavior of the complete three-layer model is examined in Section 5.5, while marking variation and the additional simulation analyses are addressed in Section 5.6.

5.3. IoT Subsystem Simulation

The IoT subnet occupies an intermediate position between the Cloud and Client subnets. Figure 4 presents the marking evolution of four selected places of this subnet (m17, m20, m22, and m25), recorded during the same single 5 s baseline simulation run of the complete model used for the Cloud analysis. The horizontal axis represents simulation time, while the vertical axis represents the marking of each monitored place.



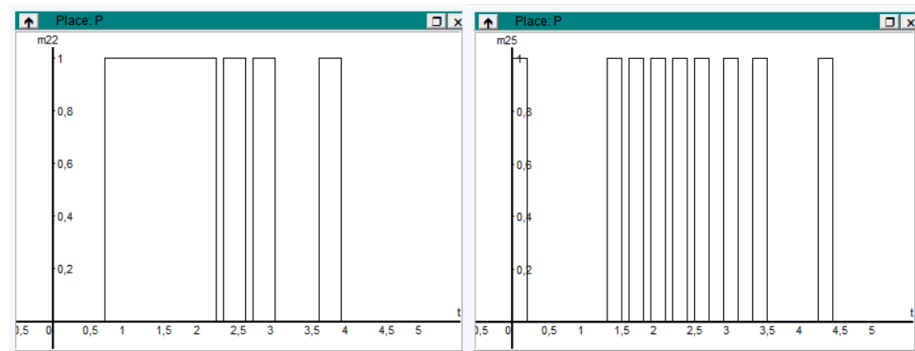


Figure 4. Marking evolution at selected places of the IoT subnet: m17, m20, m22, and m25.

Observation. During this simulation run, the four monitored IoT places exhibit different marking patterns. The marking of m17 increases progressively in a stepwise manner throughout the simulated interval. In contrast, m20, m22, and m25 repeatedly alternate between markings of 0 and 1, although the timing and duration of their marked states differ. Thus, within this run, the marking evolution is not uniform across the selected IoT places.

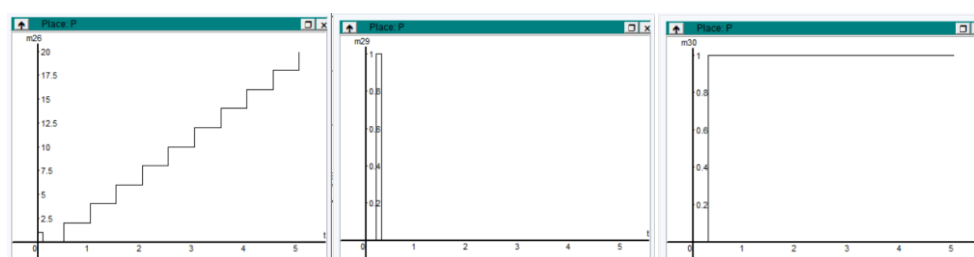
Interpretation. The different marking patterns observed at m17, m20, m22, and m25 show that the selected IoT places follow different state evolutions during this run. The progressive increase observed at m17 differs from the repeated changes between 0 and 1 observed at m20, m22, and m25. Within the Petri-net model, these changes reflect the firing of transitions and the structural connections between places and transitions. This behavior is consistent with the global state equation (Equation (2)), which relates changes in the system marking to the firing of transitions.

The use of Petri nets for representing dynamic interactions in IoT systems is supported by recent research. Yang et al. use high-level Petri nets to model interactions among IoT services and to detect functional and quality-of-service conflicts arising from their concurrent use [36]. This provides methodological support for the Petri-net formalism adopted here, without implying that the specific behavior observed in Figure 4 has been established by that study.

Implication. The different marking patterns observed at m17, m20, m22, and m25 show that the selected IoT places follow different state evolutions during this run. The progressive increase observed at m17 differs from the repeated changes between 0 and 1 observed at m20, m22, and m25. Within the Petri-net model, these changes reflect the firing of transitions and the structural connections between places and transitions.

5.4. Client Subsystems Simulation

The three Client subnets represent the user-facing part of the proposed architecture and correspond to the three stakeholder roles defined in Section 3: the learner (Client_1), the teacher (Client_2), and the institutional administrator (Client_3). Figure 5 presents the marking evolution of selected places from the three Client subnets during the same single baseline simulation run used for the Cloud and IoT analyses.



(a)

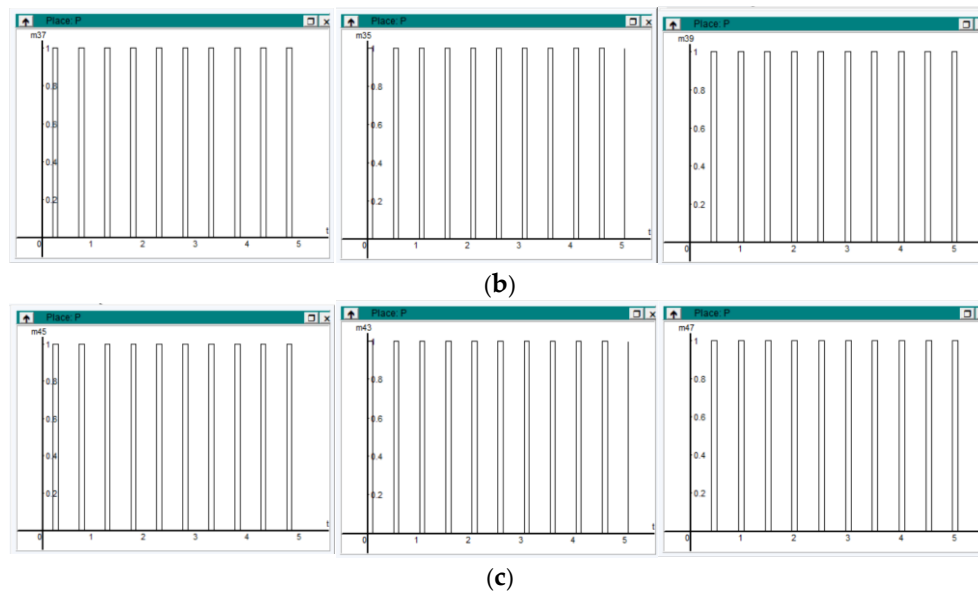


Figure 5. Marking evolution at selected places of the three Client subnets: (a) Client_1—m26, m29, and m30; (b) Client_2—m35, m37, and m39; (c) Client_3—m43, m45, and m47.

Observation. During this simulation run, the monitored places of the three Client subnets exhibit different marking patterns. For Client_1, the marking of m26 increases progressively in a stepwise manner throughout the simulated interval, while m29 is marked only briefly near the beginning of the run and m30 reaches a marking of 1 shortly after the start and remains at this value. For Client_2, m35, m37, and m39 repeatedly alternate between markings of 0 and 1. A similar repeated pattern is observed at m43, m45, and m47 for Client_3. Thus, activity is observed in all three Client subnets, but the selected places do not exhibit identical marking behavior.

Interpretation. The observed differences indicate that the selected places do not participate in the simulated Client processes in the same way. Their marking evolution depends on their position and connections within the corresponding Petri-net subnet. These patterns describe the internal behavior of the model and should not be interpreted as differences in activity or performance among the three stakeholder roles.

The separation of these roles into distinct formal components is consistent with previous applications of Petri nets in educational systems. Hammami and Mathkour proposed an adaptive e-learning architecture combining multi-agent technology with adaptive e-learning and used Object Petri Nets to formally specify message exchanges and their synchronization among the participating agents [37]. This provides a relevant example of the use of Petri nets for representing communication and synchronization in an adaptive e-learning environment, while the particular three-role organization used here remains specific to the proposed architecture.

Implication. The Client subnet simulation confirms that the three stakeholder categories defined in the architecture are represented within the executable and exhibit marking activity during the baseline simulation run. Their separation into Client_1, Client_2, and Client_3 preserves the functional distinctions established in Section 3.

The result is therefore relevant primarily at the architectural and behavioral levels. It characterizes their behavior within the simulated model, rather than establishing differences in their performance or educational outcomes. The combined behavior of the Client, IoT, and Cloud subnets is examined in Section 5.5.

5.5. Cross-Layer Information-Flow Analysis

The results presented in Sections 5.2–5.4 can be considered together to examine the simulated behavior across the three interconnected subnets of the proposed architecture. The Cloud, IoT, and Client subnets are represented as distinct components of the same formal model, with the IoT subnet providing the connection between client-level activities and the Cloud processing environment.

Observation. The results presented in Figures 3–5 show marking activity across the Cloud, IoT, and Client subnets during the same baseline simulation run. Although the selected places exhibit different marking patterns, state changes are observed at all three architectural levels. Taken together, these results provide a cross-layer view of the behavior of the complete Cloud-IoT-Client model.

Interpretation. The cross-layer behavior corresponds to the architectural organization established in Section 3 and to the formal representation developed in Section 4. The Cloud subnet represents the four core processing subsystems, the IoT subnet represents the intermediate technological layer, and Client_1, Client_2, and Client_3 represent the learner, teacher, and institutional administrator, respectively. At the formal level, Equation (2) relates changes in the system marking to transition firings, while Equation (3) represents the feedback path through which processed information can return toward the operational level. Together, the simulation and formal representation describe the dynamic interaction among the three architectural levels.

The use of Petri-net modeling and simulation for analyzing dynamic behavior in distributed computing environments is also supported by related research. Vardakis et al. [38] present a Petri-net-based framework for modeling and simulating resource-scheduling policies in an Edge Cloud Continuum, using Petri nets to model possible system states and to simulate resource-scheduling decisions in a multi-cluster environment [38]. Their study demonstrates the use of Petri-net modeling for analyzing scheduling behavior in a distributed computing setting. The specific Cloud-IoT-Client organization considered in the present study, however, remains part of the proposed architecture.

A distinction must also be made between observing information-flow behavior and performing quantitative performance evaluation. Sanahmadi et al. [39] propose a stochastic reward-net (SRN) model for modeling and quantitatively evaluating energy consumption in IoT systems [39]. Their approach demonstrates how a formal stochastic model can be used to investigate energy consumption before system implementation. Similarly, Melo et al. [40] use stochastic Petri nets to evaluate the dependability and energy-driven availability of an autonomous LoRaMesh router node and perform sensitivity analysis [40]. These studies illustrate how stochastic Petri-net models can support quantitative evaluation when explicit metrics, model parameters, and evaluation procedures are defined. The present simulation does not provide direct quantitative performance measures of this type (Section 5.8).

Implication. The combined results support the interpretation of the proposed architecture as an interconnected three-layer model in which the Cloud, IoT, and Client subnets have distinct but connected functions. The Cloud provides the central processing environment, the IoT subnet mediates the exchange between the Cloud and Client levels, and the three Client subnets represent the defined stakeholder categories. The combined analysis therefore provides a behavioral link between the architectural structure and its formal Petri-net representation.

As throughout Sections 5.2–5.4, these results do not provide direct quantitative performance measures or formally verify the properties defined in Section 4 (Section 5.8). The cross-layer results therefore provide the basis for the more focused examination of the formal and behavioral properties considered in Section 5.6.

5.6. Simulation and Formal Analysis

5.6.1. Timing Sensitivity Analysis

To further examine the influence of the transition-delay parameters on the simulated behavior of the model, a timing sensitivity experiment was performed in Visual Object Net++. Four configurations were considered: the original model (baseline) and three modified configurations in which the transition delays of the Cloud, IoT, or Client subnet were increased. In each modified configuration, the transition delays of the corresponding subnet were increased by 25%, while the remaining model parameters were kept at their baseline values. Each configuration was simulated three times, with a simulation time of 5 s for each run. The same representative places (m0, m17, m26, m35, and m43) were monitored in all configurations. The final markings obtained at the end of each simulation are reported in Table 3.

Table 3. Final marking values obtained in the timing sensitivity analysis.

Scenario	Run	Modified Delays	m0	m17	m26	m35	m43
Baseline	1	Original values	1	23.33	20	1	1
Baseline	2	Original values	1	25	25	1	1
Baseline	3	Original values	2	26.66	26.66	1	1
Cloud-delay increase	1	Cloud transition delays increased by 25%	2	26.66	26.66	1	1
Cloud-delay increase	2	Cloud transition delays increased by 25%	1	26.66	26.66	1	1
Cloud-delay increase	3	Cloud transition delays increased by 25%	2	23.33	20	1	1
IoT-delay increase	1	IoT transition delays increased by 25%	2	25	25	1	1
IoT-delay increase	2	IoT transition delays increased by 25%	2	26.66	26.66	2	2
IoT-delay increase	3	IoT transition delays increased by 25%	2	23.33	23.33	1	1
Client-delay increase	1	Client transition delays increased by 25%	2	12	12	1	1
Client-delay increase	2	Client transition delays increased by 25%	1	14	14	1	1
Client-delay increase	3	Client transition delays increased by 25%	1	12	12	1	1

The results show that the effect of the delay increase depends on the subsystem in which it is introduced. For the Cloud- and IoT-delay configurations, the final markings of m17 and m26 remained within ranges comparable to those observed in the baseline runs. A more pronounced change was observed when the Client transition delays were increased. In all three Client-delay runs, m17 and m26, reached final markings between 12 and 14, whereas the corresponding baseline values ranged from 23.33 to 26.66 for m17 and from 20 to 26.66 for m26. The remaining monitored places showed only small differences across the configurations: m0 remained between 1 and 2, while m35 and m43 remained at 1 in all runs except one IoT-delay run, in which both reached 2.

These results indicate that, under the investigated configuration and the selected simulation time, increasing the Client transition delays produced a more pronounced change in the monitored final markings than the corresponding delay increases applied to the Cloud or IoT subnets. The experiment is not intended as a direct measurement of end-to-

end latency or throughput. Instead, it provides a controlled timing sensitivity analysis based on the final markings of the Petri-net model.

5.6.2. Concurrent Workload Analysis

To examine the behavior of the complete model under increased concurrent activity, an additional experiment was performed in Visual Object Net++. The initial markings of the three Client entry places (m26, m35, and m43) were increased from one token in the baseline configuration to three and five tokens, representing moderate and higher concurrent workloads. Here, each token represents one unit of activity entering the corresponding Client subnet. Accordingly, the 1-, 3-, and 5-token configurations represent increasing levels of concurrent Client-side activity at each of the three Client entry places, rather than direct counts of individual users. In the educational scenario, these activities are associated with the corresponding Client role: learner, teacher, or institutional administrator. The model structure and transition delays were kept unchanged, and each configuration was simulated three times, with a simulation time of 5 s for each run. Two monitoring sets were used: Set A included places m0, m17, m26, m35, and m43, while Set B included m0, m10, m17, and m20. The final markings are reported in Table 4.

Table 4. Final marking values obtained in the concurrent workload analysis.

Monitoring Set	Place	Baseline (1)			Moderate (3)			Higher (5)		
		Run 1	Run 2	Run 3	Run 1	Run 2	Run 3	Run 1	Run 2	Run 3
A	m0	1	1	2	2	2	1	2	2	2
	m17	23.33	25	26.66	25	25	25	25	25	25
	m26	20	25	26.66	25	25	25	30	30	30
	m35	1	1	1	3	2	2	4	4	4
	m43	1	1	1	3	2	2	4	4	4
B	m0	1	2	1	2	1	1	1	2	1
	m10	2	4	3	4	1	1	3	4	3
	m17	26.66	23.33	23.33	26.66	26.66	25	25	25	26.66
	m20	0	0	0	0	0	0	0	0	0

The results show that the effect of increasing the initial workload was not uniform across the monitored places. In Set A, the final marking of m26 increased from 20–26.66 in the baseline runs to 30 in all runs under the higher workload. The final markings of m35 and m43 also increased, reaching 4 under the higher workload. In contrast, m17 remained within a similar range across the three workload configurations, while m0 remained between 1 and 2. A similar behavior was observed for the common places monitored in Set B. The final marking of m17 remained between 23.33 and 26.66, while m0 remained between 1 and 2. Among the additional places in Set B, m10 varied between 1 and 4 without a consistent increase with workload, whereas the final marking of m20 was zero in all runs.

Overall, the results show different final marking patterns across the monitored places as the concurrent Client workload was increased. While higher final markings were observed at the Client entry places, the monitored internal places did not show a corresponding systematic increase over the tested workload levels. This experiment provides a controlled assessment of the complete model under increased workload; however, the observed markings should not be interpreted as direct measurements of end-to-end latency, throughput, or scalability.

5.6.3. Analysis of Marking Variation

Observation. The simulated marking evolution shown in Figures 3–5 is not uniform across the selected places of the Cloud, IoT, and Client subnets. Some places exhibit relatively regular changes, whereas others show more pronounced stepwise variations. Such differences can be observed within the Cloud subnet (Figure 3), the IoT subnet (Figure 4), and the three Client subnets (Figure 5).

Interpretation. The different marking patterns indicate that the places do not participate in the information-flow process in an identical manner. Their evolution is influenced by the connections among places and transitions, transition-enabling conditions, and the transition-delay parameters represented in the model.

Implication. Places showing more pronounced marking variations can therefore be identified as monitored places exhibiting higher marking variability under the selected simulation configuration. These observations alone do not imply that the corresponding locations constitute system bottlenecks. The controlled experiment presented in Section 5.6.1 complements this analysis by examining the response of selected markings to sub-system-specific changes in transition delays.

5.6.4. Formal-Property Analysis

The cross-layer behavior analyzed in Section 5.5 provides evidence of the correspondence between the architectural structure and the execution of the Petri-net model. A complementary analysis concerns the formal properties introduced in Section 4, namely conservativeness (Equation (4)), deadlock avoidance (Equation (5)), and buffer limitation (Equation (6)).

Observation. The simulated execution shows successive transition firings and marking activity across the Client, IoT, and Cloud subnets. These executions provide evidence of reachable states under the selected simulation configuration. However, selected simulation runs do not constitute an exhaustive reachability analysis and do not, by themselves, establish liveness or the absence of deadlocks for all reachable markings.

Interpretation. Reachability concerns the markings that can be obtained from the initial marking through transition firings, while boundedness concerns whether the number of tokens remains finite over the reachable state space. Boundedness is therefore relevant to the buffer-limitation condition introduced in Equation (6). The conservativeness condition in Equation (4) addresses the preservation of the defined token quantity, while the buffer-limitation condition in Equation (6) addresses the restriction of token accumulation. Deadlock avoidance in Equation (5) is related to the absence of execution states from which the model cannot continue. These properties are established concepts in Petri-net analysis and can be examined using appropriate formal verification methods [30].

For the proposed architecture, these properties provide formal criteria for analyzing the admissible evolution of information through the Cloud, IoT, and Client subnets. The Visual Object Net++ simulation should not, by itself, be interpreted as a formal verification of conservativeness, boundedness, liveness, or deadlock-freeness, since selected simulation traces do not exhaust the reachable state space. Reachability-based graph methods can be used for the formal analysis of such properties, as demonstrated by Gu et al. [41] for bounded Petri nets [41]. To complement the simulation-based observations, Section 5.7 therefore applies state-space analysis to a reduced GSPN representation of a representative Cloud-IoT processing and feedback cycle using GreatSPN. This complementary analysis examines the reachable state space, dead markings, place boundedness, and reachability of the response state. The resulting properties apply specifically to the reduced verification model and are not generalized to the complete Petri net shown in Figure 2.

Implication. The Visual Object Net++ results provide qualitative evidence of information-flow behavior and reveal differences in marking evolution among the places of the complete model. At the formal level, the conditions defined in Section 4 provide criteria for analyzing token preservation, boundedness, deadlock avoidance, and the admissible evolution of the system. Because these properties cannot be established exhaustively from simulation traces alone, the reduced-model analysis presented in Section 5.7 provides a complementary formal verification step, while preserving a clear distinction between properties observed or verified for the reduced model and those that remain unverified for the complete architecture.

5.7. Reduced-Model Formal Verification and Steady-State Analysis with GreatSPN

5.7.1. Reduced GSPN Model and Formal Verification

To complement the simulation-based analysis performed in Visual Object Net++, a reduced Generalized Stochastic Petri Net (GSPN) was constructed and analyzed using GreatSPN, a software environment for the modeling, validation, and analysis of stochastic Petri-net models [64]. The purpose of this complementary model is not to reproduce the complete Petri net shown in Figure 2, but to isolate a representative cyclic processing path and subject it to state-space analysis. This distinction is necessary because the Visual Object Net++ simulations presented in Sections 5.2–5.6 describe the behavior of the complete architecture under the selected simulation configuration, whereas exhaustive properties of the reachable state space cannot be inferred from individual simulation traces alone.

The reduced model isolates the fundamental cross-layer processing cycle of the proposed architecture: a request is transmitted through the IoT layer, processed at the Cloud level, converted into a response, and returned through the feedback path. This cycle was selected because it preserves the interaction among the three architectural levels and the basic request-processing-response-feedback sequence while remaining sufficiently compact for exhaustive state-space analysis. The reduced model therefore provides a tractable formal representation of the core processing cycle, for which the complete reachable state space can be examined. The reduction intentionally omits additional branches and concurrent interactions of the complete model. Consequently, the formal properties established for the reduced GSPN cannot exclude deadlocks or other behaviors that may arise from those omitted interactions in the complete architecture. Its purpose is therefore to formally examine this representative cross-layer mechanism rather than to reproduce or verify the complete Petri-net model shown in Figure 2.

It comprises four places, $P_{Request}$, P_{IoT} , P_{Cloud} , $P_{Response}$ and four transitions, T_{IoT} , T_{Cloud} , $T_{Response}$ and $T_{Feedback}$. The processing cycle is represented as $P_{Request} \rightarrow T_{IoT} \rightarrow P_{IoT} \rightarrow T_{Cloud} \rightarrow P_{Cloud} \rightarrow T_{Response} \rightarrow P_{Response} \rightarrow T_{Feedback} \rightarrow P_{Request}$.

The initial marking consists of one token in $P_{Request}$, T_{IoT} , T_{Cloud} and $T_{Feedback}$ are exponentially timed transitions, while is defined as an immediate transition. The latter represents response generation as an event without a positive residence time in the corresponding intermediate state. The feedback arc from $T_{Feedback}$ to $P_{Request}$ closes the processing cycle and provides a reduced representation of the feedback mechanism defined for the architecture in Section 4.

Figure 6 shows the reduced GSPN implemented in GreatSPN, representing the request-processing-response-feedback cycle described above. The four places represent successive states of this cycle, while the transitions model the movement of the token through the IoT and Cloud processing stages, response generation, and feedback to the initial state.

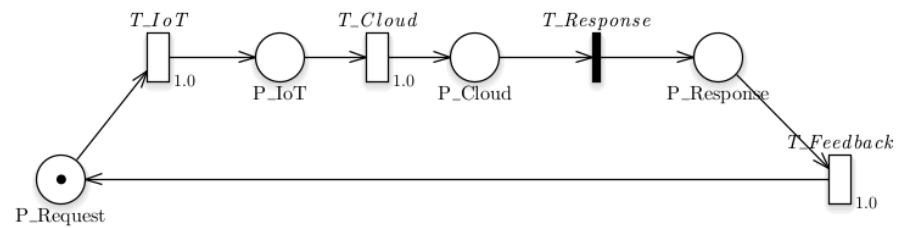


Figure 6. Reduced GSPN representation of the Cloud-IoT processing and feedback cycle implemented in GreatSPN.

As shown in Figure 7, the complete reachability graph of the reduced GSPN contains four reachable markings, comprising three tangible markings and one vanishing marking (Figure 7a). No dead markings were identified, and GreatSPN reports the initial marking as a home state (Figure 7b). Thus, for this reduced model, no reachable marking terminates the processing cycle, and the initial marking can be reached again from every reachable marking. The presence of one vanishing marking is consistent with the use of the immediate transition $T_{Response}$, since the marking enabling this transition has zero residence time in the underlying stochastic process.

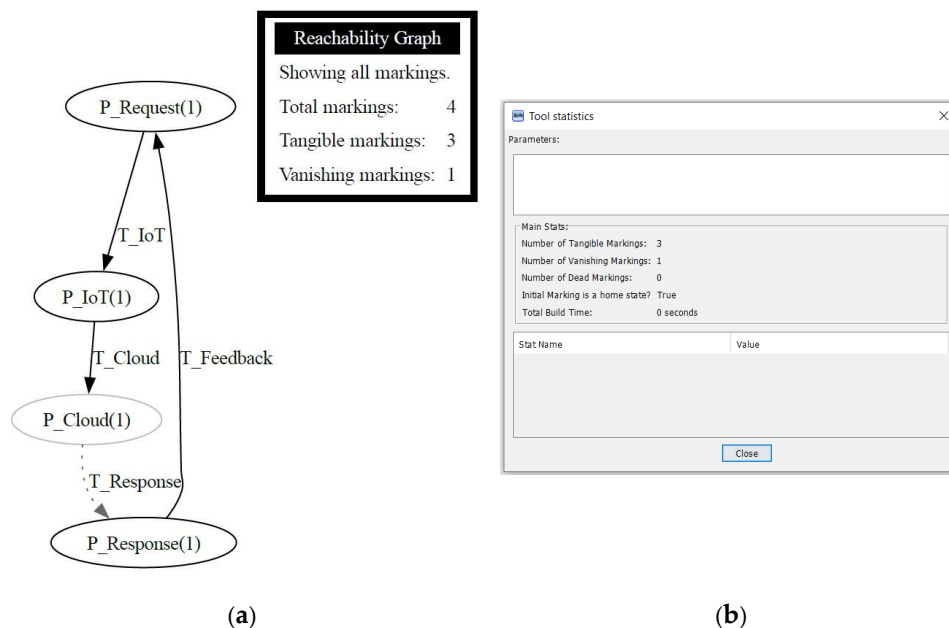


Figure 7. Reachability analysis of the reduced GSPN in GreatSPN: (a) complete reachability graph comprising four reachable markings, including three tangible and one vanishing marking; (b) reachability-graph statistics showing no dead markings and confirming the initial marking as a home state.

Structural analysis further identified one place semiflow covering all four places and established that all places are bounded. For the single-token configuration used here, these results are consistent with the cyclic circulation of the token through the reduced processing path and exclude unbounded token accumulation within this model. They also provide a formal counterpart, at reduced-model level, to the boundedness and buffer-limitation considerations introduced in Section 4 and discussed in Section 5.6.

Reachability of the response state was additionally examined through Computation Tree Logic (CTL) model checking. The following property was evaluated:

$$E[(P_{Response} = 0)U(P_{Response} \neq 0)]$$

where E is the existential path quantifier, indicating that at least one execution path satisfying the specified condition exists, and U is the CTL “until” operator. Accordingly, the formula specifies that there exists an execution path along which $P_{Response}$ remains unmarked ($P_{Response} = 0$) until a marking in which $P_{Response}$ contains a token ($P_{Response} \neq 0$) is reached. GreatSPN evaluated this property as *true*, confirming that the response state is reachable from the initial marking in the reduced GSPN model.

Taken together, the reachability graph, structural, and CTL results provide complementary formal evidence for the behavior of the selected processing cycle. Specifically, the reduced model has a finite reachable state space, contains no dead markings, has bounded places, can return to its initial marking, and admits a path reaching the response state. These conclusions apply exclusively to the reduced GSPN analyzed in GreatSPN and should not be interpreted as an exhaustive verification of the complete Cloud-IoT-Client Petri net shown in Figure 2. The reduced model is subsequently used in Section 5.7.2 to examine how controlled variations in transition rates affect its steady-state behavior.

5.7.2. Steady-State Analysis Under Transition-Rate Variations

Following the state-space analysis presented in Section 5.7.1, the reduced GSPN model was further examined through steady-state analysis in GreatSPN. The purpose of this analysis was to determine how controlled changes in the rates of the exponentially timed transitions affect the long-run distribution of the token among the processing states. This analysis complements the qualitative marking variations observed in the complete Visual Object Net++ model by providing numerical steady-state results for the reduced processing cycle. It does not constitute a measurement of latency, throughput, or response rate in the complete Cloud-IoT-Client architecture.

A baseline scenario was first defined by assigning a rate of 1.0 to each of the three exponentially timed transitions, T_{IoT} , T_{Cloud} , and $T_{Feedback}$. $T_{Response}$ remained an immediate transition in all scenarios. Under the baseline configuration, the steady-state occupancies of $P_{Request}$, P_{IoT} , and $P_{Response}$ were each approximately 0.333, while the occupancy of P_{Cloud} was 0. The latter result is consistent with the immediate firing of $T_{Response}$: the marking associated with P_{Cloud} is vanishing and therefore has no positive steady-state residence probability.

To examine the effect of a reduced transition rate, three additional scenarios were analyzed. In each scenario, the rate of one exponentially timed transition was reduced from 1.0 to 0.5, while the rates of the other two exponentially timed transitions remained equal to 1.0. The resulting steady-state occupancies are reported in Table 5.

Table 5. Steady-state place occupancies of the reduced GSPN under controlled transition-rate variations.

Scenario	T_{IoT}	T_{Cloud}	$T_{Feedback}$	$P_{Request}$	P_{IoT}	P_{Cloud}	$P_{Response}$
Baseline	1.0	1.0	1.0	0.333	0.333	0.000	0.333
Reduced IoT rate	0.5	1.0	1.0	0.500	0.250	0.000	0.250
Reduced Cloud rate	1.0	0.5	1.0	0.250	0.500	0.000	0.250
Reduced feedback rate	1.0	1.0	0.5	0.250	0.250	0.000	0.500

The results show a consistent shift in steady-state occupancy toward the place immediately preceding the transition whose rate was reduced. When the rate of T_{IoT} was reduced to 0.5, the occupancy of $P_{Request}$ increased from approximately 0.333 to 0.500, while those of P_{IoT} and $P_{Response}$ decreased to 0.250. When the rate of T_{Cloud} was reduced to 0.5, the highest occupancy shifted to P_{IoT} , which reached 0.500. Similarly, reducing the rate of $T_{Feedback}$ to 0.5 increased the occupancy of $P_{Response}$ to 0.500. In all four scenarios, P_{Cloud} retained zero steady-state occupancy because $T_{Response}$ remained immediate.

For an exponentially timed transition with rate λ , the mean firing delay is $1/\lambda$. Therefore, within the time scale of the reduced model, changing a transition rate from 1.0 to 0.5 increases its mean firing delay from 1 to 2 model time units. The steady-state results reflect this change: the model spends a larger proportion of its long-run behavior in the place preceding the reduced-rate transition. Thus, the analysis provides a controlled illustration of how rate variations at different stages of the processing cycle affect the distribution of system activity.

These results also complement the marking-variation observations reported in Section 5.6. In the complete Visual Object Net++ model, differences among places were identified qualitatively from their marking evolution under the selected simulation configuration. The GreatSPN analysis provides a distinct quantitative perspective at reduced-model level by showing that a controlled change in an individual transition rate produces a corresponding measurable change in steady-state occupancy. The two analyses therefore address different aspects of model behavior and should not be interpreted as a quantitative sensitivity analysis of the complete architecture.

The transition rates used in these scenarios are model parameters selected for comparative analysis and are not derived from empirical measurements of a deployed Cloud-IoT educational system. Consequently, the values in Table 5 should be interpreted as results of a controlled parametric experiment rather than as estimates of real system performance. Nevertheless, the consistent response across the three rate-reduction scenarios demonstrates that the reduced GSPN captures the expected relationship between transition timing and state occupancy and provides a quantitative complement to the behavioral analysis of the complete model. Formal verification and quantitative performance evaluation of the complete Cloud-IoT-Client model, as well as empirical calibration of its transition rates, remain subjects for further investigation.

5.8. Architectural Interpretation and Discussion

The simulation provides a qualitative basis for interpreting the behavior of the proposed architecture. The observed variations in marking evolution show different state evolutions within the Cloud, IoT, and Client subnets under the temporal configuration represented by the transition-delay parameters. These results describe how system activity evolves under the selected simulation configuration but do not constitute direct quantitative performance measurements.

The performance measures introduced in Section 4 can be related to the complete model once empirical timing data are available. For a defined processing path, the latency of an activity can be expressed as

$$L = t_{response} - t_{entry}, \quad (7)$$

where t_{entry} and $t_{response}$ denote the times at which the same activity enters the processing path and produces the corresponding response. Throughput over an observation interval can be expressed as

$$X = \frac{N_{completed}}{T_{obs}}, \quad (8)$$

where $N_{completed}$ denotes the number of activities completed during the observation interval T_{obs} . Applying these measures to the complete model would require recording the corresponding entry and response events and calibrating the transition-delay parameters using empirical processing and communication times. Equations (7) and (8) provide a basis for future quantitative evaluation and are not derived from the marking results reported in the present study.

The concurrent workload analysis also shows differences between the Client entry places and the monitored places in the Cloud and IoT subnets. As the initial Client workload increased, higher final markings were observed at the Client entry places, particularly at m26, while the monitored Cloud and IoT places did not show a similar systematic increase. This behavior suggests that activity may accumulate at the Client entry places under the tested conditions, but the marking results alone are not sufficient to identify a specific bottleneck in the Cloud or IoT subnets. In a real implementation, this behavior could be examined further using empirical measurements under different workload conditions. If processing or communication constraints are identified, edge processing or cloud load balancing could be considered as possible ways to distribute the workload.

The complete Visual Object Net++ model and the reduced GreatSPN model provide complementary levels of analysis. The former shows marking activity and state evolution across the Cloud, IoT, and Client subnets and supports the correspondence between the architectural organization introduced in Section 3 and its executable Petri-net representation.

The reduced GSPN complements this behavioral analysis through formal state-space and steady-state evaluation of a representative Cloud-IoT processing and feedback cycle. For this reduced model, the results establish bounded state evolution, absence of dead markings, reachability of the response state, and a consistent relationship between transition-rate variation and steady-state place occupancy.

These two analyses therefore have different scopes. Visual Object Net++ provides behavioral evidence for the complete architecture, whereas GreatSPN provides formal and quantitative results for a reduced representative model. The GreatSPN results are not extended to the complete Cloud-IoT-Client Petri net. The limitations of both levels of analysis are summarized in Section 5.9.

5.9. Limitations and Future Research

The present study has several limitations. First, although the reduced GSPN provides numerical steady-state results, latency, throughput, and response rate have not been quantitatively evaluated for the complete Cloud-IoT-Client model. Moreover, the transition rates used in the GreatSPN analysis are controlled scenario parameters rather than values derived from empirical measurements.

Second, the formal results obtained with GreatSPN apply only to the reduced processing and feedback cycle and cannot be generalized to the complete Petri-net model. Exhaustive verification of the formal conditions defined in Section 4 and of related properties such as liveness and boundedness therefore remains necessary at the complete-model level.

Third, the marking variations identified in Sections 5.2–5.6 characterize the behavior of the complete model under the selected simulation configurations and should not be interpreted as direct performance or bottleneck measures. The timing sensitivity and concurrent workload analyses introduced in Section 5.6 provide quantitative results in terms of marking behavior, but they do not constitute direct measurements of system-level performance. The rate-variation analysis in Section 5.7.2 provides complementary quantitative evidence for the reduced GSPN model.

Finally, the study remains model-based and does not include validation against a deployed Cloud-IoT educational system. The present results should therefore be viewed as a model-level evaluation of the proposed architecture, while its behavior and scalability under real operating conditions require validation in a deployed environment. Future research should extend formal verification and quantitative performance analysis to the complete model, calibrate model parameters using empirical data, and validate the resulting behavior against measurements from a real implementation.

6. Conclusions

This study proposed a Cloud-IoT-Client architecture for educational environments and developed a timed Petri-net model to represent its dynamic behavior. The architecture integrates three interconnected levels—the Cloud, IoT, and Client levels—and distinguishes three client roles: learner, teacher, and institutional administrator. Within the business-language learning instantiation considered in this study, the Cloud level comprises four processing subsystems: the Domain Terminology Repository, Adaptive Learning Engine, Monitoring and Assessment Module, and Administrative and Reporting Module. The formal model represents the interactions among these components through state evolution, transition-delay parameters, and feedback mechanisms.

The complete Petri-net model was implemented and simulated using Visual Object Net++. The simulation results show marking activity and state evolution within the Cloud, IoT, and Client subnets and reveal differences in marking variation among the selected places. All three client configurations exhibit marking activity within the simulated model, while the cross-layer analysis provides a combined view of the behavior observed across the three architectural levels. These results provide behavioral evidence of the correspondence between the conceptual architecture and its executable Petri-net representation. They should not, however, be interpreted as exhaustive verification of the complete model or as direct measurements of system performance.

Additional controlled experiments examined the complete model under subsystem-specific timing variations and increased concurrent Client workloads. These analyses provided quantitative results in terms of marking behavior, without being interpreted as direct measurements of latency, throughput, or scalability.

To complement the simulation-based analysis, a reduced GSPN representing a Cloud-IoT processing and feedback cycle was constructed and analyzed using GreatSPN. State-space analysis of this reduced model established bounded state evolution, absence of dead markings, and reachability of the response state. The steady-state analysis further showed that reducing the rate of an exponentially timed transition increased the long-run occupancy of the place immediately preceding that transition. Across the controlled rate-variation scenarios, this relationship was observed consistently, providing quantitative evidence of how local timing parameters affect the distribution of activity within the reduced processing cycle. These results apply specifically to the reduced GSPN and do not constitute formal verification or quantitative performance evaluation of the complete Cloud-IoT-Client model.

Taken together, the Visual Object Net++ and GreatSPN analyses provide complementary evidence at different levels of the proposed model. The former enables behavioral examination of the complete architecture, whereas the latter provides formal state-space and quantitative steady-state results for a representative reduced cycle. This combined analysis strengthens the connection between the architectural description and its formal representation while preserving the distinction between simulation-based observations, reduced-model verification, and actual system performance.

Several limitations remain. Latency, throughput, and response rate have not been quantitatively evaluated for the complete architecture, and the transition rates used in the

GreatSPN experiments are controlled model parameters rather than values calibrated from empirical measurements. The timing sensitivity and concurrent workload analyses provide controlled assessments of marking behavior under subsystem-specific delay changes and increased Client workloads, but they do not constitute direct performance, scalability, or bottleneck analyses. Future work should therefore extend formal verification and quantitative performance analysis to the complete Petri-net model, calibrate transition parameters using empirical data, consider a broader range of timing and workload conditions, and validate the model against measurements obtained from a deployed Cloud-IoT educational environment.

The present results therefore provide a basis for further development, empirical validation, and complete-model verification of Cloud-IoT educational architectures.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/iot7040085/s1>, File S1: Petri-net models and associated files used for the simulation and formal analyses presented in the manuscript (Visual Object Net++ and GreatSPN).

Author Contributions: Conceptualization, E.M.C. and L.L.B.; methodology, E.M.C. and L.L.B.; software, E.M.C. and L.L.B.; validation, E.M.C., L.L.B. and V.T.D.; formal analysis, E.M.C.; investigation, L.L.B.; resources, E.M.C. and L.L.B.; data curation, L.L.B. and E.M.C.; writing—original draft preparation, L.L.B. and E.M.C.; writing—review and editing, L.L.B.; visualization, E.M.C.; supervision, V.T.D.; project administration, E.M.C. and V.T.D.; funding acquisition—not applicable. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The models and associated files supporting the results presented in this study are available in the Supplementary Materials.

Acknowledgments: During the preparation of this manuscript, the authors used ChatGPT (OpenAI) and Claude (Anthropic) as supporting tools for language refinement, with the aim of improving the clarity and readability of the text. As the manuscript was developed and revised over an extended period, during which these tools were updated, different versions available at the respective stages were used rather than a single software version. The tools were also used to assist in reviewing how the results were presented and discussed, particularly with regard to clarity and consistency throughout the manuscript. All AI-assisted content was reviewed and revised by the authors before being incorporated into the manuscript, with the final content reflecting the authors' own scientific judgment.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviation

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
CTL	Computation Tree Logic
DSPN	Deterministic and Stochastic Petri Net
GSPN	Generalized Stochastic Petri Net
IoT	Internet of Things
IoT-A	Internet of Things Architecture
LoRaWAN	Long Range Wide Area Network
LPWAN	Low-Power Wide-Area Network
QoS	Quality of Service
RFID	Radio-Frequency Identification
SRN	Stochastic Reward Net

References

1. Bauer, M.; Boussard, M.; Bui, N.; De Loof, J.; Magerkurth, C.; Meissner, S.; Nettsträter, A.; Stefa, J.; Thoma, M.; Walewski, J.W. IoT Reference Architecture. In *Enabling Things to Talk*; Bassi, A., Bauer, M., Fiedler, M., Kramp, T., van Kranenburg, R., Lange, S., Meissner, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 163–211. https://doi.org/10.1007/978-3-642-40403-0_8.
2. *IEEE Std 2413-2019*; IEEE Standard for an Architectural Framework for the Internet of Things (IoT). IEEE: Piscataway, NJ, USA, 2019. <https://doi.org/10.1109/IEEESTD.2020.9032420>.
3. Rasyid, R.E.; Aisa, S.; Rizal, A. Global Trends and Contributions of Internet of Things (IoT) Research in Education: A Bibliometric Analysis. *Mimb. Ilmu* **2024**, *29*, 500–506. <https://doi.org/10.23887/mi.v29i3.89589>.
4. Zeeshan, K.; Hämäläinen, T.; Neittaanmäki, P. Internet of Things for Sustainable Smart Education: An Overview. *Sustainability* **2022**, *14*, 4293. <https://doi.org/10.3390/su14074293>.
5. Todorov, T.; Vela, P. Internet of Things in Education. In *Science Series “Innovative STEM Education”, Volume 05*; Institute of Mathematics and Informatics—Bulgarian Academy of Sciences: Sofia, Bulgaria, 2023; pp. 193–200, ISSN 2683-1333. <https://doi.org/10.55630/STEM.2023.0522>.
6. Qureshi, K.N.; Naveed, A.; Kashif, Y.; Jeon, G. Internet of Things for education: A smart and secure system for schools monitoring and alerting. *Comput. Electr. Eng.* **2021**, *93*, 107275. <https://doi.org/10.1016/j.compeleceng.2021.107275>.
7. Prasetya, L.A.; Rofiudin, A.; Herwanto, H.W. Implementation of Internet of Things (IoT) in Education: A Systematic Literature Review. *J. Educ. Comput. Appl. JECA* **2025**, *2*, 1–45. <https://doi.org/10.69693/jeca.v2i1.19>.
8. Spaho, E.; Çiço, B.; Shabani, I. IoT Integration Approaches into Personalized Online Learning: Systematic Review. *Computers* **2025**, *14*, 63. <https://doi.org/10.3390/computers14020063>.
9. Kuldeep, S.F.; Faustin, P.B. IoT for Smart Classroom in Improving Teaching and Learning Approach. *Asia-Afr. J. Educ. Res. AAJER* **2022**, *1*, 86–108. Available online: <https://journals.iapaar.com/index.php/aajer/article/view/44> (accessed on 20 August 2026).
10. Botta, A.; De Donato, W.; Persico, V.; Pescapé, A. Integration of Cloud computing and Internet of Things: A survey. *Future Gener. Comput. Syst.* **2016**, *56*, 684–700. <https://doi.org/10.1016/j.future.2015.09.021>.
11. Al-Emran, M.; Malik, S.I.; Al-Kabi, M.N. A Survey of Internet of Things (IoT) in Education: Opportunities and Challenges. In *Toward Social Internet of Things (SIoT): Enabling Technologies, Architectures and Applications*; Hassanien, A.E., Bhatnagar, R., Khalifa, N.E.M., Taha, M.H.N., Eds.; Springer: Cham, Switzerland, 2020; Volume 846, pp. 197–209. https://doi.org/10.1007/978-3-030-24513-9_12.
12. de Lima, C.C.; Schlemmer, E.; Morgado, L. Internet of Things and Education: A Systematic Literature Review. *Res. Soc. Dev.* **2020**, *9*, e6039119674. <https://doi.org/10.33448/rsd-v9i11.9674>.
13. Jurva, R.; Matinmikko-Blue, M.; Niemela, V.; Nenonen, S. Architecture and Operational Model for Smart Campus Digital Infrastructure. *Wirel. Pers. Commun.* **2020**, *113*, 1437–1454. <https://doi.org/10.1007/s11277-020-07221-5>.
14. Nie, X. Constructing Smart Campus Based on the Cloud Computing Platform and the Internet of Things. In *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*; Atlantis Press: Dordrecht, Netherlands, 2013; pp. 1576–1578. <https://doi.org/10.2991/iccsee.2013.395>.
15. Fraga-Lamas, P.; Celaya-Echarri, M.; Lopez-Iturri, P.; Castedo, L.; Azpilicueta, L.; Aguirre, E.; Suárez-Albela, M.; Falcone, F.; Fernández-Caramés, T. Design and Experimental Validation of a LoRaWAN Fog Computing Based Architecture for IoT Enabled Smart Campus Applications. *Sensors* **2019**, *19*, 3287. <https://doi.org/10.3390/s19153287>.
16. Pexyeen, T.; Saraubon, K.; Nilsook, P. IoT, AI and Digital Twin for Smart Campus. In *Proceedings of the 2022 Research, Invention, and Innovation Congress: Innovative Electricals and Electronics (RI2C)*, Bangkok, Thailand, 4–5 August 2022; pp. 160–164. <https://doi.org/10.1109/RI2C56397.2022.9910286>.
17. Haggag, M.; Oulefki, A.; Amira, A.; Kurugollu, F.; Mushtaha, E.; Soudan, B.; Hamad, K.; Fofou, S. Integrating Advanced Technologies for Sustainable Smart Campus Development: A Comprehensive Survey of Recent Studies. *Adv. Eng. Inform.* **2025**, *66*, 103412. <https://doi.org/10.1016/j.aei.2025.103412>.
18. Koukaras, C.; Hatzikranielis, E.; Mitsiaki, M.; Koukaras, P.; Tjortjis, C.; Stavrinides, S. Revolutionising Educational Management with AI and Wireless Networks: A Framework for Smart Resource Allocation and Decision-Making. *Appl. Sci.* **2025**, *15*, 5293. <https://doi.org/10.3390/app15105293>.
19. Elbertsen, M.; Kok, H.; Salimi, N. Designing the Future Smart Campus: Integrating Key Elements to Enhance User Experience. *J. Sci. Technol. Policy Manag.* **2025**, *16*, 117–137. <https://doi.org/10.1108/JSTPM-10-2024-0414>.

20. Arsenijević, D.; Arsenijević, J.; Tegeltija, S.; Zhang, X.; Ostojić, G.; Stankovski, S. Assessing Internet of Things Readiness on University Campuses: A Smart Campus-Oriented Approach. *IoT* **2026**, *7*, 39. <https://doi.org/10.3390/iot7020039>.
21. Burunkaya, M.; Duraklar, K. Design and Implementation of an IoT-Based Smart Classroom Incubator. *Appl. Sci.* **2022**, *12*, 2233. <https://doi.org/10.3390/app12042233>.
22. Dai, Z.; Zhang, Q.; Zhao, L.; Zhu, X.; Zhou, D. Cloud-Edge Computing Technology-Based Internet of Things System for Smart Classroom Environment. *ijET* **2023**, *18*, 79–96. <https://doi.org/10.3991/ijet.v18i08.28299>.
23. Sharma, T.; Aarthy, S.L. An Automatic Attendance Monitoring System Using RFID and IoT Using Cloud. In Proceedings of the 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, India, 19 November 2016; pp. 1–4. <https://doi.org/10.1109/GET.2016.7916851>.
24. Dhir, S.; Maheshwari, A.; Sharma, S. Improving Teaching-Learning Through Smart Classes Using IoT. In *Enhancing Education with Intelligent Systems and Data-Driven Instruction*; Bhatia, M., Mushtaq, M.T., Eds.; IGI Global Scientific Publishing: Hershey, PA, USA, 2024; pp. 107–131. <https://doi.org/10.4018/979-8-3693-2169-0.ch006>.
25. Jayousi, S.; Lucattini, P.; Petti, L.; Bruni, F.; Mucchi, L. Smart Learning by Design: A Framework for IoT-Driven Adaptive Classrooms and Inclusive Education. *Educ. Sci.* **2025**, *15*, 1338. <https://doi.org/10.3390/educsci15101338>.
26. Quevedo Piratova, D.A.; Sánchez Duarte, M.M.; Chiappe, A.; Sáez Delgado, F.M. Intelligent Classrooms: How AI and IoT Can Reshape Learning Spaces. *Eur. J. Educ.* **2025**, *60*, e70000. <https://doi.org/10.1111/ejed.70000>.
27. Khashan, O.A.; Alamri, S.; Alomoush, W.; Alsmadi, M.K.; Atawneh, S.; Mir, U. Blockchain-Based Decentralized Authentication Model for IoT-Based E-Learning and Educational Environments. *Comput. Mater. Contin.* **2023**, *75*, 3133–3158. <https://doi.org/10.32604/cmc.2023.036217>.
28. Andriulo, F.C.; Fiore, M.; Mongiello, M.; Traversa, E.; Zizzo, V. Edge Computing and Cloud Computing for Internet of Things: A Review. *Informatics* **2024**, *11*, 71. <https://doi.org/10.3390/informatics11040071>.
29. Boca, L.L.; Ciortea, E.M.; Boghean, C.; Begov-Ungur, A.; Boghean, F.; Dădârlat, V.T. An IoT System Proposed for Higher Education: Approaches and Challenges in Economics, Computational Linguistics, and Engineering. *Sensors* **2023**, *23*, 6272. <https://doi.org/10.3390/s23146272>.
30. Murata, T. Petri Nets: Properties, Analysis and Applications. *Proc. IEEE* **1989**, *77*, 541–580. <https://doi.org/10.1109/5.24143>.
31. Ahmad, F.; Chaudhry, M.T.; Jamal, M.H.; Sohail, M.A.; Gavilanes, D.; Vergara, M.M.; Ashraf, I. Formal Modeling and Analysis of Security Schemes of RPL Protocol Using Colored Petri Nets. *PLoS ONE* **2023**, *18*, e0285700. <https://doi.org/10.1371/journal.pone.0285700>.
32. Fortas, A.; Kerkouche, E.; Chaoui, A. High-Level Petri Nets Based Approach for Designing and Analyzing IoT Systems. *Internet Things* **2025**, *34*, 101783. <https://doi.org/10.1016/j.iot.2025.101783>.
33. Balogh, Z.; Turčáni, M.; Magdin, M. The Possibilities of Using Petri Nets for Realization of a Universal Model of Educational Process. In Proceedings of the 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI), San Francisco, CA, USA, 14–16 August 2013; pp. 162–169. IEEE. <https://doi.org/10.1109/IRI.2013.6642468>.
34. Balogh, Z.; Kuchárik, M. Predicting Student Grades Based on Their Usage of LMS Moodle Using Petri Nets. *Appl. Sci.* **2019**, *9*, 4211. <https://doi.org/10.3390/app9204211>.
35. Khamparia, A.; Pandey, B. Threat Driven Modeling Framework Using Petri Nets for E-Learning System. *SpringerPlus* **2016**, *5*, 446. <https://doi.org/10.1186/s40064-016-2101-0>.
36. Yang, R.; Wu, M.; Gui, X.; Chen, H. Intelligent Conflict Detection of IoT Services Using High-Level Petri Nets. *Complex Intell. Syst.* **2024**, *10*, 3789–3817. <https://doi.org/10.1007/s40747-024-01349-8>.
37. Hammami, S.; Mathkour, H. Adaptive E-Learning System Based on Agents and Object Petri Nets (AELS-A/OPN). *Comput. Appl. Eng. Educ.* **2015**, *23*, 170–190. <https://doi.org/10.1002/cae.21587>.
38. Vardakis, C.; Dimolitsas, I.; Spatharakis, D.; Dechouniotis, D.; Zafeiropoulos, A.; Papavassiliou, S. A Petri Net-based Framework for Modeling and Simulation of Resource Scheduling Policies in Edge Cloud Continuum. *Simul. Model. Pract. Theory* **2025**, *141*, 103098. <https://doi.org/10.1016/j.simpat.2025.103098>.
39. Sanahmadi, A.; Abdollahi Azgomi, M.; Goudarzi, S. An SRN-based Model for Quantitative Evaluation of IoT Quality Attributes. *Internet Things* **2023**, *23*, 100894. <https://doi.org/10.1016/j.iot.2023.100894>.
40. Melo, C.; Barros, L.; Rezende, J.; Araujo, J. Towards Dependable IoT in Remote Environments: A Stochastic Petri Net Model for Energy-Harvesting LoRaMesh Router Nodes. *J. Reliab. Intell. Environ.* **2026**, *12*, 8. <https://doi.org/10.1007/s40860-026-00270-w>.
41. Gu, C.; Ma, Z.; Li, Z. Liveness and Deadlock-Freeness Verification and Enforcement in Bounded Petri Nets Using Basis Reachability Graphs. *Automatica* **2024**, *164*, 111625. <https://doi.org/10.1016/j.automatica.2024.111625>.

42. Andrade, E.; Nogueira, B.; de Farias Júnior, I.; Araújo, D. Performance and Availability Trade-Offs in Fog–Cloud IoT Environments. *J. Netw. Syst. Manag.* **2021**, *29*, 2. <https://doi.org/10.1007/s10922-020-09570-9>.
43. Nguyen, T.A.; Fe, I.; Brito, C.; Kaliappan, V.K.; Choi, E.; Min, D.; Lee, J.W.; Silva, F.A. Performability Evaluation of Load Balancing and Fail-over Strategies for Medical Information Systems with Edge/Fog Computing Using Stochastic Reward Nets. *Sensors* **2021**, *21*, 6253. <https://doi.org/10.3390/s21186253>.
44. Bogarín, A.; Cerezo, R.; Romero, C. A Survey on Educational Process Mining. *WIREs Data Min. Knowl. Discov.* **2018**, *8*, e1230. <https://doi.org/10.1002/widm.1230>.
45. van der Aalst, W.M.P. The Application of Petri Nets to Workflow Management. *J. Circuits Syst. Comput.* **1998**, *8*, 21–66. <https://doi.org/10.1142/S0218126698000043>.
46. Qiu, S. Improving Performance of Smart Education Systems by Integrating Machine Learning on Edge Devices and Cloud in Educational Institutions. *J. Grid Comput.* **2024**, *22*, 41. <https://doi.org/10.1007/s10723-024-09755-5>.
47. Ahaidous, K.; Tabaa, M.; Hachimi, H. Towards IoT-Big Data Architecture for Future Education. *Procedia Comput. Sci.* **2023**, *220*, 348–355. <https://doi.org/10.1016/j.procs.2023.03.045>.
48. Badshah, A.; Rehman, G.U.; Farman, H.; Ghani, A.; Sultan, S.; Zubair, M.; Nasralla, M.M. Transforming Educational Institutions: Harnessing the Power of Internet of Things, Cloud, and Fog Computing. *Future Internet* **2023**, *15*, 367. <https://doi.org/10.3390/fi15110367>.
49. Zhu, Z.-T.; Yu, M.-H.; Riezebos, P. A Research Framework of Smart Education. *Smart Learn. Environ.* **2016**, *3*, 4. <https://doi.org/10.1186/s40561-016-0026-2>.
50. Țălu, M. Exploring IoT Applications for Transforming University Education: Smart Classrooms, Student Engagement, and Innovations in Teacher and Student-Focused Technologies. *Bul. Ilm. Sarj. Tek. Elektro* **2025**, *7*, 9–29. <https://doi.org/10.12928/biste.v7i1.12361>.
51. Satu, M.S.; Roy, S.; Akhter, F.; Whaiduzzaman, M. IoLT: An IoT Based Collaborative Blended Learning Platform in Higher Education. In *Proceedings of the 2018 International Conference on Innovation in Engineering and Technology (ICIET 2018)*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6. <https://doi.org/10.1109/CIET.2018.8660931>.
52. Kuner, C.; Bygrave, L.A.; Docksey, C.; Drechsler, L. (Eds.) *The EU General Data Protection Regulation (GDPR): A Commentary*; Oxford University Press: Oxford, UK, 2020. <https://doi.org/10.1093/oso/9780198826491.001.0001>.
53. Liu, Q.; Khalil, M. Understanding Privacy and Data Protection Issues in Learning Analytics Using a Systematic Review. *Br. J. Educ. Technol.* **2023**, *54*, 1715–1747. <https://doi.org/10.1111/bjet.13388>.
54. Slade, S.; Prinsloo, P. Learning Analytics: Ethical Issues and Dilemmas. *Am. Behav. Sci.* **2013**, *57*, 1510–1529. <https://doi.org/10.1177/0002764213479366>.
55. International Telecommunication Union. *Recommendation ITU-T Y.2060: Overview of the Internet of Things*; International Telecommunication Union: Geneva, Switzerland, 2012.
56. Dake, D.K.; Bada, G.K.; Dadzie, A.E. Internet of Things (IoT) Applications in Education: Benefits and Implementation Challenges in Ghanaian Tertiary Institutions. *J. Inf. Technol. Educ. Res.* **2023**, *22*, 311–338. <https://doi.org/10.28945/5183>.
57. Molloy, M.K. Performance Analysis Using Stochastic Petri Nets. *IEEE Trans. Comput.* **1982**, *C-31*, 913–917. <https://doi.org/10.1109/TC.1982.1676110>.
58. Ajmone Marsan, M.; Balbo, G.; Conte, G.; Donatelli, S.; Franceschinis, G. *Modeling with Generalized Stochastic Petri Nets*; John Wiley & Sons: New York, NY, USA, 1995; ISBN 0-471-93059-8.
59. Haas, P.J. *Stochastic Petri Nets: Modeling, Stability, Simulation*; Springer Series in Operations Research; Springer: New York, NY, USA, 2002; ISBN 978-0-387-95445-5.
60. Ficili, I.; Giacobbe, M.; Tricomi, G.; Puliafito, A. From Sensors to Data Intelligence: Leveraging IoT, Cloud, and Edge Computing with AI. *Sensors* **2025**, *25*, 1763. <https://doi.org/10.3390/s25061763>.
61. Afrin, S.; Rafa, S.J.; Kabir, M.; Farah, T.; Alam, M.S.B.; Lameesa, A.; Ahmed, S.F.; Gandomi, A.H. Industrial Internet of Things: Implementations, Challenges, and Potential Solutions across Various Industries. *Comput. Ind.* **2025**, *170*, 104317. <https://doi.org/10.1016/j.compind.2025.104317>.
62. López Camacho, V.; de la Guía, E.; Orozco-Barbosa, L.; Olivares, T. WIoTED: An IoT-Based Portable Platform to Support the Learning Process Using Wearable Devices. *Electronics* **2020**, *9*, 2071. <https://doi.org/10.3390/electronics9122071>.
63. Chen, Q.; Wang, Z.; Su, Y.; Fu, L.; Wei, Y. Educational 5G Edge Computing: Framework and Experimental Study. *Electronics* **2022**, *11*, 2727. <https://doi.org/10.3390/electronics11172727>.

64. Amparore, E.G.; Balbo, G.; Beccuti, M.; Donatelli, S.; Franceschinis, G. 30 Years of GreatSPN. In *Principles of Performance and Reliability Modeling and Evaluation: Essays in Honor of Kishor Trivedi on His 70th Birthday*; Fiondella, L., Puliafito, A., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 227–254. https://doi.org/10.1007/978-3-319-30599-8_9.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.