Wireless Personal Communications (2023) 133:1693–1747
https://doi.org/10.1007/s11277-023-10842-1

Modelling, Simulation, and Performance Analysis of Intra-Vehicular Heterogeneous Networks

Guillermo Funes[1] · **Mario Siller**[1] · **Jorge Horta**[1]

Accepted: 21 December 2023 / Published online: 27 January 2024

Abstract
Vehicles today use a variety of network segments operated by different technologies and protocols within the car (CAN, LIN, Automotive Ethernet, MOST, FlexRay, etc.) to exchange data between different control modules, sensors, and actuators. The exchange of information between other network domains (heterogeneous networks) is enabled through various interconnection points called gateways/bridges. The resulting performance depends on its interconnection structure, network segment traffic aggregation scheme, and medium access technique. Although protocols such as CAN, LIN, FlexRay, and Ethernet have been used in network design for some time, performance modeling and analysis are still needed given the variety of traffic types and sources, new application limitations, and especially the lack of formal verification of network performance for different network scenarios and configurations. This paper presents an end-to-end throughput and delay performance analysis for a reference intra-vehicular network scenario. These models have been validated through simulations in which high correlation values were obtained from 98.7400 to 99.9999, with a low mean square error. The validation cases show that for different LIN, CAN, and Ethernet network configurations, the performance threshold values defined for most current vehicle applications are preserved. However, if the network configuration is modified, the proposed analytical models can be used to formally verify the corresponding performance and delay changes and thus validate whether or not the application requirements are met.

Keywords Intra-vehicular networks · Heterogeneous networks · Performance modeling and analysis · Teletraffic theory · Network simulation

✉ Mario Siller
mario.siller@cinvestav.mx

Guillermo Funes
mfunes8@gmail.com

Jorge Horta
jorge.horta@cinvestav.mx

[1] CINVESTAV Unidad Guadalajara, Del Bosque Av. 1145, 45019 Zapopan, Jalisco, Mexico

1 Introduction

The internal vehicle/automobile network better known as intra-vehicular network (IVN), is an area of study of great importance in terms of network performance analysis. As technology advances in the automotive industry, the number of sensors, actuators, and ECUs (Electronic Control Units) used in vehicles has increased. For this reason, cars became a set of systems that collaborate to perform more complex tasks, a system of systems. These require the different component systems to communicate with each other while meeting bandwidth and latency requirements. Additionally, in-vehicle communications are crucial in developing *x-by-wire* capabilities in future vehicles [1].

The above results in the emergence of different intra-vehicular network protocols such as CAN (Controller Area Network), LIN (Local Interconnect Network), MOST (Media Oriented Systems Transport), FlexRay, Automotive Ethernet, and others, which define different types of functions to meet the requirements of the different car domains [2]. Therefore, internal vehicle networks are classified as heterogeneous communication networks comprising two or more network segments/domains, each defined by a single protocol.

The increase in the number of ECUs generates problems related to the weight, cost, complexity, and reliability required by the cables and connectors involved in connecting the ECUs. For this reason, bus-based networks are proposed to communicate the different ECUs within the vehicle [3]. These networks require appropriate protocols to communicate over the shared bus.

In general, ECUs communicate with and control the essential components of the vehicle system (sensors and actuators). With these characteristics, it is possible to consider them as a cyber-physical system (CPS), which is an integration of cybernetic components (computers) that interact with physical elements of an environment through an actuator [4]. Important examples of CPS are intelligent vehicles and roads, part of Intelligent Transportation Systems (ITS). Given the current trend of research and development of ITSs, there is a growing interest in vehicular communications. As stated in [5], vehicular communication is crucial to improving road safety and transport efficiency.

Currently, vehicular applications are considered in various IoT (Internet of Things) domains, highlighting the area of smart cities. In [6], the Internet of Vehicles (IoV) is described as a new technology for developing transport systems in Smart Cities. For this purpose, vehicular networks can be designed using communications between vehicles (V2V or vehicle-to-vehicle transmission), between vehicles and road infrastructure (V2I or vehicle-to-infrastructure communication), and other elements of the surrounding environment (V2X or vehicle-to-everything communication).

The state-of-the-art usually focuses on proposing techniques to improve the performance of vehicular networks or develop new applications in these environments. For instance, the work [7] suggests the AS-DTMAC (Active Signaling-based TDMA MAC) protocol to improve the delay generated by the TDMA (Time Division Multiple Access) schemes used in a VANET (Vehicular Ad-Hoc Network); In [8] the authors propose a strategy aimed at helping drivers inside parking lots in a smart parking application. The vehicle can be seen as a "thing" with which the user can interact socially. The concept, architecture, and enabling technologies of the social Internet of vehicles are addressed in [9]. The authors conceive the vehicle, specifically its sensors, as part of their layered architecture's "physical world layer". Furthermore, they describe the vehicle architecture as a four-layer model: the physical layer, in-vehicle communication layer, processing layer, and application layer.

Previous works are mainly oriented to inter-vehicular communications; however, the sources of vehicle traffic are located in intra-vehicular networks; thus, their performance analysis in the context of IoV continues to be relevant. In the design of these heterogeneous intra-vehicular network architectures to share data between different network domains or protocols, the use of communication bridges/gateways is required. The performance analysis must consider these elements to estimate a transmission's end-to-end delay and throughput. These two QoS (Quality of Service) performance metrics are essential for the design of current and new vehicle applications (safety and non-safety) such as those presented at [10].

This task can be supported by analytical tools such as network performance models. This work focuses on developing analytical models that evaluate the network's performance to carry out the specification and formal verification of the requirements of vehicular applications associated with its communications.

The rest of this document is organized as follows. Section 2 describes some of the most common protocols used within intra-vehicular networks. Section 3 discusses layer two interconnection bridges. Section 4 presents homogeneous network models (single protocol). Section 5 discusses heterogeneous network models (two or more protocols). Results and validation through simulation for homogeneous and heterogeneous network models are presented in Sect. 6. Finally, the document concludes in Sect. 7.

2 Intra-Vehicular Network Protocols

Over the years, intra-vehicular networks have become heterogeneous networks. New network requirements have led to new wired and wireless protocols. However, legacy technologies are still relevant for vehicle networks. The authors in [11] note that even though the wireless network could reduce wiring in the vehicle, they still require connection to a power source. Additionally, security is a significant challenge for wireless networks. Therefore, wired technology, specifically Ethernet, would be a trend for vehicle networks. This change will occur gradually because an instant complete replacement of vehicle components would not be practical. The authors also claimed that Ethernet (as a backbone) would coexist with legacy technology for a few years.

The work [12] presents a recent state-of-the-art review of commonly used protocols and technologies. This review includes both wired and wireless technologies. The wired technologies reviewed include LIN, CAN, FlexRay, and MOST. A summary of the characteristics of each protocol is presented. LIN is a low-level communication system commonly used as a low-cost alternative for low-data-rate vehicle sub-subsystems. For higher-requirement applications, the CAN network is used. CAN become the most widely used wired network for vehicles. This is because CAN meets the requirements of non-stringent real-time in-vehicle systems with an event-driven approach. The FlexRay network is implemented for those subsystems and applications with higher data rates and strict time requirements. Finally, it is mentioned that MOST would be used for multimedia and telemetric applications.

The work [13] presents a performance evaluation of different wireless sensor networks inside the vehicle. The author notes that IVWSN (Intra-Vehicle Wireless Sensor Networks) integrated the sensor into vehicles, reduced assembly, and maintenance costs, as well as helping to reduce fuel consumption. The authors present UWB (Ultra-Wide Band) channel models for propagation inside the engine and under the chassis scenarios. They also

proposed different system configurations and performance analyses for each. Simulations validate the results of the study. At the end of the article, it is concluded that the IVWNS technology is not mature enough and does not provide sufficient reliability. The authors also stated that full adoption of IVWSN is not currently feasible.

The work [14] reviews the protocols, challenges, and solutions in the vehicle. The author divides the car into four subsystems: powertrain, chassis, body, and infotainment. It is mentioned that CAN is implemented for body and powertrain domains. LIN, which is primarily cost-saving oriented, is used for low-speed communications. The FlexRay protocol is used as a specialized control in the chassis as a communications backbone. Finally, MOST, the most expensive network, is used for the infotainment domain. The authors highlight that CAN stands out for its low-cost, high-reliability network. LIN is cost-effective for connecting sensors and actuators when high bandwidth is not required. FlexRay offers confidence to meet the reliability requirements of security systems. MOST is a high-speed protocol specialized in multimedia systems. It is also observed that although the MOST bandwidth is greater than the CAN, LIN, and FlexRay bandwidths, its higher cost makes it unfeasible for various subsystems. The authors conclude the article by stating that Automotive Ethernet suits the upcoming in-vehicle networking requirements.

According to references [11–14], legacy technologies are still relevant for the in-vehicle network but with a tendency to use Ethernet. This should be reflected in the study scenario: the backbone network and most of the networks that demand high bandwidth are supported by Ethernet, while the low-requirement networks are defined using LIN and CAN technologies; this is because the cost reduction continues to be an interest from automobile manufacturing. Automotive applications can be placed in different categories or classes. The Society of Automotive Engineers (SAE) [15, 16] has classified these applications into Class A, B, C, and D depending on their critical limitations in real-time and reliability. These classes are defined according to the following characteristics:

- **Class A**: network capacity of 10 Kbps or lower (low-speed network) and applied in features such as body and comfort
- **Class B**: network capacity between 10 Kbps and 125 Kbps (medium speed network) and applied in general information transfer, such as emission data and instrumentation.
- **Class C**: network capacity greater than 125 Kbps (high-speed network) and applied for real-time control such as traction control, brake by wire, etc.
- **Class D**: network capacity greater than 1 Mbps and applied for real-time safety data transmission such as the braking system, engine control, etc.

A comparison of different protocols belonging to each class is presented in Table 1:

Table 1 SAE in-vehicle network categories based on network capacity, purpose, and latency QoS metric

Class	Bandwidth	Protocol	Purpose	Latency
A	Less than 10 Kbps	LIN	Sensor/actuator control	Wide windows (<150ms)
B	10–125 Kbps	LIN, CAN	Information sharing	Varying window
C	125 Kbps–1Mbps	CAN	Real-time control	Narrow windows
D	Greater than 1Mbps	Automotive-Ethernet MOST Flexray	Real-time control and information sharing	Narrow window

In addition to the SAE automotive application classification, a different categorization can be made considering the type of services involved. According to [10], there are two main categories of application services: (i) security services and (ii) non-security-related services. Some vehicle safety services include vehicle health warnings, vehicle type warnings, traffic hazard warnings, and dynamic vehicle warnings. Non-safety-related services are mainly associated with traffic management and infotainment. Each type of service has different use cases, as shown in Table 2. The requirements for each use case can be expressed in terms of communication mode, security, reliability, minimum frequency of periodic messages, maximum latency, and technologies that could be involved (the Protocol Stack).

Table 2 shows that for most of the use cases, the communication technologies that could be involved are CAN, LIN, and Automotive Ethernet.

As indicated in [17], there are new protocols such as VaN (Vehicle Area Network), LVDS (Low-Voltage Differential Signaling), and even new versions of existing protocols such as TTCAN (Time-Triggered CAN), CANFD (CAN with flexible data rate) vehicle networks have been used. However, these are not widely adopted by OEMs. The authors also conjecture that today's most commonly used technologies (LIN, CAN, FlexRay, MOST, and Ethernet) will remain relevant for years.

There are other classifications in which more classes are identified. For instance, in [18], three main categories are defined: active road safety applications, traffic efficiency and management applications, and infotainment applications. Active road safety applications are aimed at reducing the probability of traffic accidents. The traffic management and efficiency applications aim to improve the flow of vehicular traffic. Finally, infotainment applications are aimed at offering some services (local or Internet): notification of points of interest, electronic commerce, media download, etc.

For this work, the classification presented in [10] defines the base scenario, shown in Fig. 6. In this sense, whenever a new use case is specified or the requirements change (modification of the network configuration), the corresponding delay and performance can be verified with the proposed analytical models discussed in the following sections and thus validate whether or not the application requirements are met.

Today, many of the vehicular applications developed are oriented toward ITS. ITS aims to integrate with other intelligent systems to exchange data or information to enable complex IoT applications. The emergence and growth of IoT systems encourage a large number of devices around the world to collect and transmit data, which is an integral part of the so-called Big Data [19]. At the same time, this data is stored and processed in the cloud, one of the main components of IoT systems.

The authors in [20] mention that IoT encompasses various application domains, including the cities sometimes referred to as Smart Cities. These cities include vehicles and other transportation systems. In [21], several application scenarios involving transportation systems are listed. A different work [22] refers to Smart Mobility as one of the most critical areas within Smart Cities. Smart Mobility impacts various dimensions of the smart city that directly or indirectly affect the citizens' quality of life. One of the Smart Mobility component groups is the ITS, which collects, stores, and processes data used to plan, implement, and evaluate Smart Mobility policies within this context.

The data collected by ITS comes primarily from vehicle internal components. As stated above, a vehicle comprises various elements that perform different functions and have additional requirements. Various communication protocols are used internally in vehicles because many components work together to perform specific tasks. Each protocol groups together a set of elements with standard requirements and functions, while more complex

Table 2 User cases requirements [10]

	Category	Use case	Maximum delay	Involved IVN technology
Safety Services	Vehicle status warning	Emergency electronic brake lights	100 ms	CAN
Safety Services	Vehicle status warning	Abnormal condition warning	100 ms	CAN
Safety Services	Vehicle type warning	Emergency vehicle warning	100 ms	CAN
Safety Services	Vehicle type warning	Slow vehicle warning	100 ms	CAN
Safety Services	Vehicle type warning	Motorcycle warning	100 ms	CAN
Safety Services	Vehicle type warning	Vulnerable road user warning	100 ms	CAN
Safety Services	Traffic hazard warning	Wrong way driving warning	100 ms	CAN
Safety Services	Traffic hazard warning	Stationary vehicle warning	100 ms	CAN
Safety Services	Traffic hazard warning	Traffic condition warning	100 ms	CAN
Safety Services	Traffic hazard warning	Signal violation warning	100 ms	CAN
Safety Services	Traffic hazard warning	Road work warning	100 ms	CAN
Safety Services	Traffic hazard warning	Decentralized floating car data	100 ms	CAN
Safety Services	Dynamic vehicle warning	Overtaking vehicle warning	100 ms	CAN
Safety Services	Dynamic vehicle warning	Lane change assistance	100 ms	CAN
Safety Services	Dynamic vehicle warning	Pre-crash sensing warning	50 ms	Ethernet
Safety Services	Dynamic vehicle warning	Cooperative glare reduction	100 ms	CAN
Non-safety Services	Traffic Management	Regulatory/contextual speed limits	N/A	LIN
Non-safety Services	Traffic Management	Traffic light optimal speed advisory	100 ms	CAN
Non-safety Services	Traffic Management	Intersection management	100 ms	CAN
Non-safety Services	Traffic Management	Cooperative flexible lane change	500 ms	LIN
Non-safety Services	Traffic Management	Electronic toil collect	500 ms	LIN
Non-safety Services	Infotainment	Point of interest notification	500 ms	LIN
Non-safety Services	Infotainment	Local electronic commerce	500 ms	LIN
Non-safety Services	Infotainment	Media download	500 ms	LIN
Non-safety Services	Infotainment	Map download and update	500 ms	LIN

tasks may require communicating elements operating with different protocols. Below, a review of some of the most common protocols within intra-vehicular systems is presented.

2.1 CAN Protocol

The CAN (Controller Area Network) protocol is a "serial communication technology used especially for exchanging reliable data between ECUs in cars" [23]. CAN is a message-oriented protocol where data is transmitted on the CAN bus. No addressed nodes are connected. In a CAN network, nodes send and receive messages based on a unique identifier.

Because several message transmissions share the bus, an arbitration mechanism is required. The CAN protocol bus access procedure is a non-destructive bit-wise arbitration [24] called Carrier Sense Multiple Access with Collision Detection and Arbitration on Message Priority (CSMA/CD+AMP).

The message priority defined in the CAN identifier frame field determines which node gains access to the bus. Multiple nodes can initiate the transmission of a frame whenever the bus is detected as idle. When a node sends a frame, it monitors and reads the bus bit value and compares it to its transmitted bit value. CAN messages are encoded using the NRZ (Non-Return to Zero) scheme using two bits defined as recessive equivalent to "1" and dominant as "0". During the arbitration process, the dominant bits overwrite the recessive value on the bus, such that the high-priority message is preserved (the priority is explained below). According to [25], the CAN bus arbitration process is shown in Fig. 1.

CAN arbitration process

A CAN message contains an 11 or 29-bit identifier ([26]) (for the standard and extended format, respectively) called the arbitration field. The highest priority message is transmitted on the bus due to the CAN arbitration scheme. The numerical value of the identifier in the arbitration field determines the priority. The lowest numerical value has the highest priority (the node with the most significant bit '0' in its frame identifier wins access). This bit-wise arbitration is non-destructive because if a node writes a recessive bit on the bus but reads a dominant bit due to an overwriting, the node ends its transmission and waits until the channel/bus becomes idle again to try to transmit.

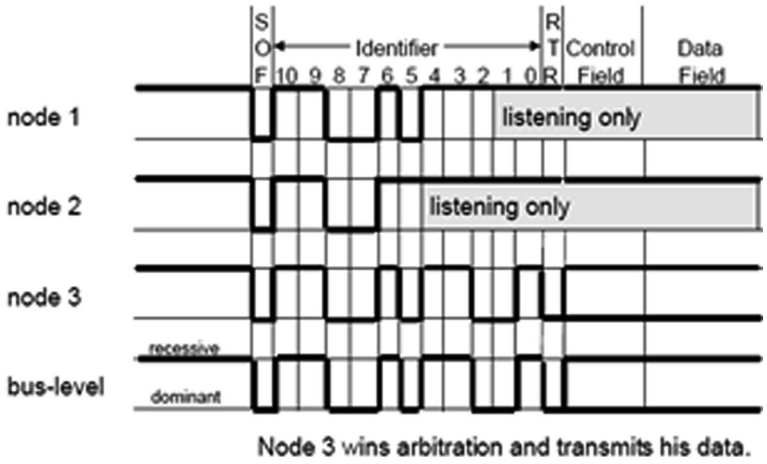


Fig. 1 CAN bus arbitration process

2.2 LIN Protocol

LIN (Local Interconnection Network) is a low-cost serial communication system. LIN employs a master–slave concept and clock synchronization to share a single-wire bus. The LIN bus is commonly used for comfort areas such as doors, seats, steering wheel, and air conditioning. The LIN Description File (LDF) is the main component and describes the entire network by defining all the properties of the network bus.

Network Basic Operation

A LIN network comprises a master node and up to 16 slave nodes. The master node controls communication in a LIN network. For example, the master node is responsible for sending sync pulses, monitoring data on the bus, switching the slave nodes to sleep/wake mode, and defining the transmission speed on the network. The operation of the LIN bus is based, as indicated above, on a single master/multiple slave concept. In a LIN bus network, a LIN node can be specified to contain one of the following two tasks:

- A master task decides the frame sequence using a schedule table.
- A slave task is responsible for transmitting requested information and waking the slave nodes from the suspended state.

It is essential to mention that usually, only the master node includes a master task and a slave task. However, each slave node contains only one slave task [27].

The LIN Master plays the most crucial role in a LIN network because it manages communications. This LIN Master is responsible for transmitting the message headers at the defined time points according to the LDF. The master node uses a schedule in the LDF to look up the dispatch time and the node identifier. When a slave node reads a message header containing an identifier that matches its own, it sends a message response containing data and a checksum. For transmitting messages on a LIN bus, the send times in the LIN Schedule must be selected to ensure sufficient time for transmitting messages. Therefore, the LIN protocol specification defines forty percent (40%) additional time according to the nominal time for transmission of a LIN message.

The LIN Schedule is organized into mini-slots; thus, the lengths of these mini-slots correspond to the fundamental time base of the LIN Schedule (for example, five milliseconds (ms)). The duration represents the smallest time resolution for processing the LIN Program by the master node. It is essential that when sending a LIN message, enough mini-slots are provided and allocated to that message to ensure its successful transmission. Furthermore, the sum of the necessary mini-slots assigned to a particular message is called the frame slot [28].

2.3 Automotive Ethernet Protocol

Automotive Ethernet was often referred to as the future of intra-vehicle networks. This protocol covers all the requirements for the intra-vehicular network and offers greater bandwidth than de-facto vehicular protocols. Furthermore, it has a variety of topologies and is suitable as a backbone bus for connecting various network protocol domains. Additionally, as noted in [29], Ethernet will be an essential component of the communication layer that will enable communication between vehicle electronics and the Internet.

The work [30] describes the electromagnetic, environmental, and electrical requirements of Ethernet. It also introduces the network fundamentals and the OSI reference model layers. In addition, a comparison between Ethernet and traditional vehicle network protocols is also presented.

The Ethernet physical layer is supported by twisted pair cables that offer bidirectional data transfers (full-duplex) with rates between 100 and 1000 Mbps. As mentioned in [31], the second layer of Ethernet communication provides essential functions for controlled data transmission. In addition to the uniform structure of messages, these include the addressing of nodes and the method of accessing the bus. All the essential functions are implemented by the Ethernet controller, which is generally an integral component of microcontrollers today. In the Ethernet bus media access mechanism, the controller listens to the physical media before sending a message (carrier sense multiple access (CSMA)). This prevents a message from being overwritten if another node on the network is already sending it, generally called a collision. If the medium is free, the Ethernet controller can begin its transmission. This work considers a backbone network based on an Ethernet bus, as it commonly occurs to communicate with internal vehicular systems. With this, it is possible to limit cables and connectors' weight, cost, complexity, and reliability. Therefore, this communication scenario is looked over and talked about below.

CSMA/CD

The Ethernet protocol refers to Layer 1 and Layer 2 of the OSI reference model. The Layer 2 bus access scheme is called Carrier Sense Multiple Access/Collision Detection (CSMA/CD). Each Ethernet controller implements the associated CSMA/CD algorithm. For the CSMA/CD procedure, all stations/ nodes monitor the channel to determine when it is idle. This is done by carrier sense circuits that operate on each Ethernet node. Collisions can occur if two or more nodes begin transmitting a frame during the same time window. Because of this, each Ethernet controller has a collision detection feature that is used to cancel the transmission each time a collision is detected. In addition, to avoid a second collision, a node tries to re-transmit based on the expiration of a random time (back-off process). Each sender must calculate this time individually.

Whenever the bus is detected busy, a node stops transmitting and begins sensing the channel. Suppose at any point the channel is seen idle or free. In that case, the sender starts sending and monitoring the channel signal to compare it to the transmitted signal and determine if a collision has occurred. Several analytical models have been proposed for this Ethernet protocol scheme (CSMA/CD) focused on performance analysis, addressing it differently. The authors in [32] proposed a mathematical delay and throughput model for a new efficient CSMA/CD-based protocol based on advanced channel slot reservation with priority access. In [33], aspects that influence latency are presented, and then an analysis of the calculation of Ethernet communication latency is proposed. Gebali [34] presents a simple CSMA/CD protocol model using teletraffic theory based on a Markov chain with three basic channel states: idle, collision, and transmission (the latter composed of n transmission states). Considering the transmission delay, propagation delay, number of nodes, constant frame size, and assuming a P-Persistent approach, the author proposed a mathematical model for performance metrics such as access probability, throughput, and others. This analytical model was used as a baseline model and was part of the end-to-end delay and throughput analysis in the heterogeneous network case study defined for the present work. Table 3 summarizes the parameters employed for the CSMA/CD model.

In [34], a Markov chain for CSMA/CD is defined, represented in the state transition diagram shown in Fig. 2.

Table 3 Description of a CSDMA/CD mathematical model variables

Variable	Description
N	Total number of Ethernet Nodes
n	The ratio of transmission delay for propagation delay
u_0	Probability of being in the idle state
u_1	Transition probability from an idle state to transmitting state
a	Packet arrival probability
Idle	Idle state
Collided	Collision state
tx_i	Transmission states
s	Steady-state probability vector

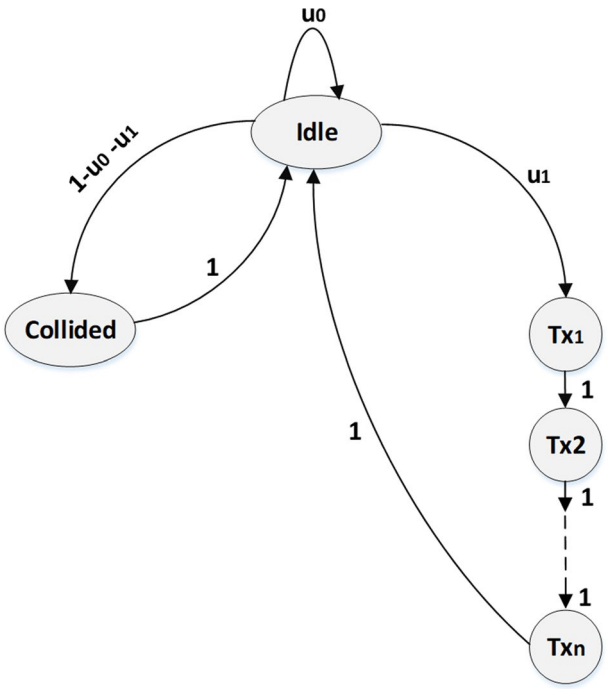


Fig. 2 CSMA/CD Markov chain [34]

This diagram defines the idle, collided, and transmission states. The transition matrix corresponding to the Markov chain in Fig. 2 is given by (Eq. 1):

$$\mathbf{P} = \begin{bmatrix} u_0 & 0 & 0 & \dots & 1 & 1 \\ u_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 1-u_0-u_1 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \tag{1}$$

In the network scenario presented in [34], N users transmit independently of any other. The probability that i users are active (try to transfer) in a given time interval is described by a binomial probability distribution (Eq. 2), where a is the probability that a single station makes a transmission during a time step:

$$u_i = \binom{N}{i} a^i (1-a)^{N-i} \tag{2}$$

The CSMA/CD model steady-state distribution vector is given by:

$$\mathbf{s} = [s_i \, s_{t1} \, s_{t2} \, ... s_{tn} \, s_c]^t \tag{3}$$

The author in [34] formulates this distribution vector at equilibrium as follows:

$$\mathbf{s} = \frac{1}{2 + u_1(n-1) - u_0} \begin{bmatrix} 1 \\ u_1 \\ u_1 \\ \vdots \\ u_1 \\ 1 - u_0 - u_1 \end{bmatrix} \tag{4}$$

In addition, he also defines values for certain performance metrics. The throughput model for CSMA/CD is given by:

$$Th_{eth} = \frac{nu_1}{2 + u_1(n-1) - u_0} \tag{5}$$

The access probability model is defined as follows:

$$\rho_a = \frac{Th_{eth}}{Na} \tag{6}$$

Finally, the average number of attempts for a successful transmission in an Ethernet network is given by:

$$\eta_a = \frac{1 - \rho_a}{\rho_a} \tag{7}$$

Using the Ethernet performance Eqs. 5–7, the Ethernet network performance can be evaluated. In addition, other metrics, such as delay, can be derived.

3 Layer 2 Bridges

In an intra-vehicular scenario, bridges, called gateways, communicate heterogeneous networks [35]. These network bridges are used to interconnect dissimilar protocols, which adds some complexity to the process.[36]. Some state-of-the-art documents related to the connection of intravehicular networks are listed below.

In [37], the author presents and describes various ways in which vehicle networks could be designed and implemented by interconnecting different network segments through bridges based on hierarchical or multi-layer physical network topology, e.g., ring, star, mesh, etc. Protocols (like CAN and LIN) are modeled, and delay and throughput

performance metrics are analyzed. Performance models are proposed for individual network segments (domains) based on channel utilization. Different hierarchical topologies (various network segments and bridges) are considered for the delay analysis. However, the analytical models of it are limited to time-triggered in-vehicle network protocols. Furthermore, the models are not validated through simulation or test-bed implementation.

In a different work [38], a gateway framework for vehicle networks based on CAN, FlexRay, and Ethernet segments is proposed. This proposal is mainly based on the network (routing) and application layers and analyzes the response time in the worst scenario case. This gateway framework was implemented through a benchmark gateway prototype. The proposal considers end-to-end response time analysis and verifies gateway latency through a test case. However, analytical models still need to be covered.

The authors in [39] proposed a bridging strategy based on a heterogeneous integrated network CAN-switched Ethernet architecture. A CAN-switched Ethernet network simulation based on OMNET ++ was developed. Simulation performance was analyzed by measuring communication latencies per device and focused on the impact of the time introduced by various CAN-Ethernet multiplexing strategies implemented on gateways. The simulation results showed that depending on its role, the gateway/bridge mapping strategy affects the message latency at the gateway compared to other bridge strategies that measure end-to-end latency. The work also included general end-to-end analytical models for latency, a CAN bus delay model, and a Ethernet layer two delay model.

Similar to [39], the authors in [40] propose a formal method to analyze and derive upper limits on end-to-end latencies for complex frame aggregation strategies in gateways. This is by capturing complex signal paths between domains that span multiple buses, gateways, and switches. A performance analysis was proposed to measure bridge delay and end-to-end performance modeling analysis based on event modeling and frame path aggregation. The model is evaluated in different network scenarios focusing on CAN to Ethernet transmission strategies. Their results show that frame aggregation can reduce the number of transmitted Ethernet frames, resulting in less load on the gateways and fewer interfering frames in the Ethernet domain, thus reducing end-to-end latency for Ethernet traffic (not between gateways). Frame aggregation can introduce a sampling delay for traffic between gateways. However, this delay can be controlled by timeouts or buffer sizes that decrease the sampling delay at the cost of increasing the load and the number of interfering Ethernet frames.

In [41], the author proposed a bridge architecture to interconnect a CAN segment, an 802.11 segment, and an 802.15.4 segment. Layer 2 performance metric models, such as delay and throughput, were presented and validated using a test-bed implementation. The performance of the bridges was analyzed considering the reception on each input interface, the frames processing for forwarding decisions, the lookup tables management, and the frames on each output interface forwarding. This performance analysis was based on queuing theory for generating bridge throughput and delay mathematical models.

Based on the reviewed state of the art, it is identified that the interconnection of the heterogeneous network (networks that operate under different network protocols, for example, LIN/CAN or Ethernet/CAN network) is more complex than the interconnection of homogeneous networks (for example, Ethernet/Ethernet or CAN/CAN). This complexity is described in terms of the associated functions for managing incoming and outgoing frames. Both translation and forwarding procedures are required to interconnect heterogeneous networks in a store-and-forward device. In contrast, the pass-through forwarding process is sufficient for the interconnection of homogeneous networks. Both

types of forwarding processes are carried out on bridges. Bridges operate in the logical link control (LLC) or medium access control (MAC) level sub-layers.

Homogeneous network interconnection is based on MAC-level bridging for different network domains that use the same frame format and employ the same data link protocol. On the other hand, the interconnection of LANs that use a different frame format and a different data link layer protocol is based on LLC-level bridging. As a result, translation is necessary when networks have different frame formats [42]. Figure 3 shows a) bridged representation for the same frame format (same protocol) and b) bridged representation for different frame formats (different protocols).

The present work's analytical models for network bridges will be based on queuing theory. This is due to its behavior as a store-and-forward device.

4 Homogeneous Network Models

This section models each network domain that made up the baseline scenario without considering the interconnection between network protocols.

4.1 CAN Protocol Delay and Throughput Model

The CAN protocol has been modeled and analyzed with different approaches. The work [43] presents a real-time framework for stochastic latency analysis of distributed CAN automotive systems. Based on periodic tasks (deterministic activation times), the authors propose a stochastic calculation to calculate the pmf (probability mass function) of the response time of CAN messages. In a different work [44], a CAN communication bus is modeled using Matlab/Simulink^{TM}. They consider the error frames to calculate response times and check the bus load.

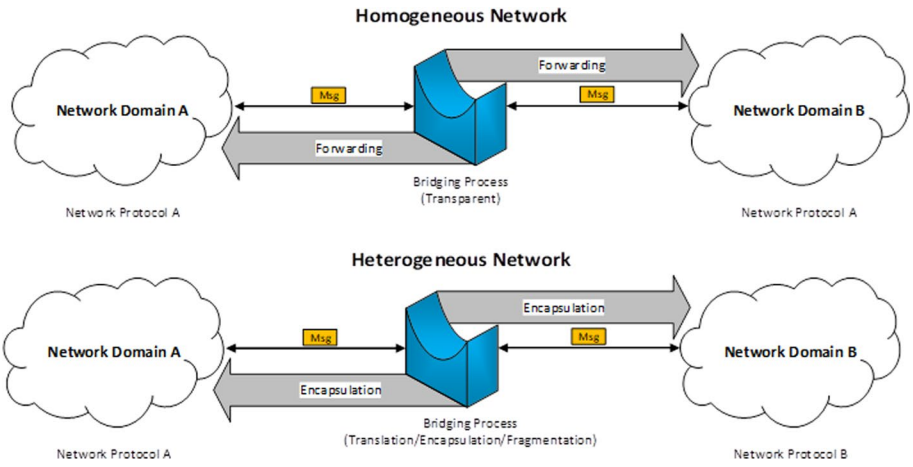


Fig. 3 Bridge representation used for interconnecting networks operating under the same and different protocols

The CAN mathematical model represents the present work's contention phase and transmission process. The CAN network is expressed as a set T of message transmissions. Concerning [41], the number of messages transmitted in a CAN network can be represented as a set M of messages. These messages can be divided into a 7-tuple$< Arb_i, Ctrl_i, Data_i, CRC_i, Ack_i, EoF_i, IFS_i >$, where:

- Arb_i: {11,29} arbitration field (11-bit Identifier for Standard frame format and 29 for Extended frame format)
- $Ctrl_i$: 10 bits control field
- $Data_i$: {0–8} bytes payload field
- CRC_i: 16 bits CRC (Cyclic Redundancy Check) field
- Ack_i: 2 bits acknowledge field
- EoF_i. 7 bits end of a frame bit field
- IFS_i: 3 bits inter-frame space

The proposed CAN model represents the transmission process in the following two stages:

1. Start of Frame (SOF) bit transmission and the arbitration process.
2. The control, payload, CRC, ACK, and EoF fields transmission.

The CAN model Markov chain, presented in Fig. 4, has a state space defined as $\psi = \{Idle, Cttn_i, Tx_i, Err_k, IFS_l, LW\}$ where:

- $1 \leq i \leq Cttn$ (Id bits sent in the bus contention phase)
- $1 \leq j \leq CTCA$ (Control, Transmission of data (Payload), CRC, ACKbits)
- $1 \leq k \leq 20$ (Error frame bits)
- $1 \leq k \leq 3$ (IFS bits (3 inter-frame space bits plus 7 end-of-frame (EoF) bits))

Referring to the CAN Markov chain (Fig. 4), the idle state is represented by the gray oval shape; the yellow states are the contention procedure; the light green oval shapes represent the control, data, CRC, ACK states and EoF in the transmission process, the green states are the IFS bits that specify the end of a CAN frame, the red oval shapes represent the error frame transmission, and the blue oval shape is the waiting/listening state.

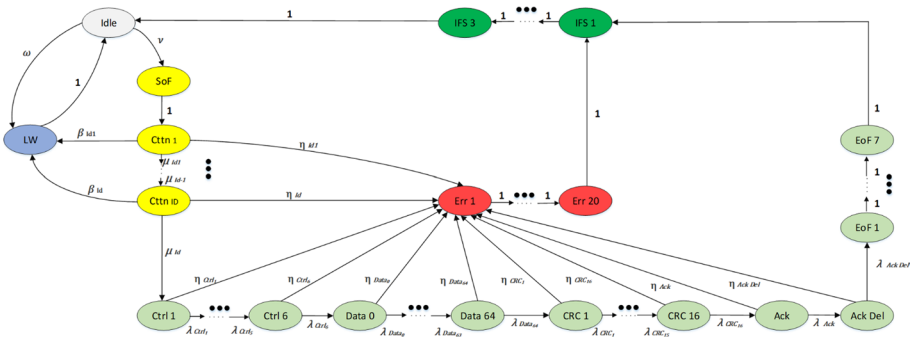


Fig. 4 CAN Layer 2 (CSMA/CD+AMP) Markov Chain

For stage one, the first SOF bit is sent. A node with a transmission frame enters this stage when the channel is detected idle with probability v or goes to the waiting/listening stage with probability ω. After that, the arbitration scheme occurs the {11 or 29 } **I d** bits of CAN messages transmitted simultaneously by a set of **n** nodes are compared. After the SOF bit, all competing nodes transition with probability 1 to the first comparison state; from here, the transition to the following comparison state with probability μ_{Id_i} occurs. If the node message identifier has a lower priority than others, that node enters the wait/listen state with probability β_{Id_i} due to a contention loss. After comparing the total Id bits, the contention winning node proceeds to Stage Two with probability $\mu_{Id_{end}}$. In this second stage, the node transmits the bits corresponding to the control, data, CRC, ACK, and EoF fields with probability λ_j. Errors can occur during any bit transmission. This happens during the contention process with probability η_{Id_i} and for the transmission process with probability η_{Tx_i}. The table 4 describes the variables used for the model.

In addition, a Markov superstate chain (illustrated in Fig. 5) is developed based on the Markov chain in Fig. 4. The super-state transition matrix was also defined to understand better how the balanced equations and delay and throughput models were formulated.

The transition matrix of the Super-state Markov chain (Fig. 5) is given by:

Table 4 Description of variables used for CAN probabilities and mathematical models

Variable	Description
n	Total number of CAN nodes
Id	Bit quantity in the arbitration field given by message Id
CTCA	Total number of bits consisting of Control, Transmission of data (Payload), CRC, ACK bits of a CAN frame
IFS	Total number of bits of Inter-frame Space (IFS) and EoF bits
F_{size}	CAN frame size
τ	The bit time
H_{size}	CAN Frame header size
Idle	Idle state
$Cttn_i$	The contention process states
Tx_j	The transmission process states
IFS_k	The final transmission states
Err_k	The transmission error states
LW	The waiting process states
v	Probability to transition from the idle state to contention state
μ_i	Probability to check the next i+1 arbitration bit
η_i	Probability that a bit error occurs in the i^{th} bit in the arbitration process and k bit in the transmission process
λ_i	Probability to transmit i^{th} CAN frame part (Control, Payload, CRC, and Ack bits)
ω	Probability to transition from Idle state to waiting state because of a busy channel
β_i	Probability to transition from the contention state to waiting state because of losing arbitration in the i^{th} arbitration bit
ψ	The vector of steady states probability

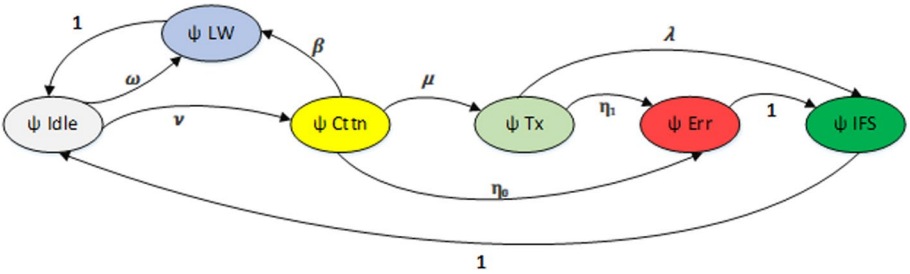


Fig. 5 CAN Layer 2 (CSMA/CD+AMP) Super states Markov Chain

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ \omega & 0 & \beta & 0 & 0 & 0 \\ \nu & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 & 0 & 0 \\ 0 & 0 & \eta_0 & \eta_1 & 0 & 0 \\ 0 & 0 & 0 & \lambda & 1 & 0 \end{bmatrix} \tag{8}$$

Loss Probability Model

As indicated in [41], the loss probability consists of the sum of all the transition probabilities from *Cttn* to Err_1 plus the sum of all the transition probabilities from Tx_j a Err_1. Furthermore, it is assumed that a transmitted bit error occurs independently of the transmission of other bits, and the transmission process follows a Poisson distribution. The loss probability model is given by:

$$Loss_p = \sum_{i=1}^{Id}(1-\mu_i) + \sum_{j=1}^{CTCA}(1-\lambda_j) \tag{9}$$

Throughput Model

The first term of the model (Eq. 10) represents the data portion of the CAN frames without the overhead. The second term denotes the correct or incorrect reception of the frame since the receiving node acknowledges positively or negatively (by modifying the bit in the ACK slot), and the transmitting node awaits this confirmation. If the receiver does not acknowledge the reception of the message, the transmitting node will try to send the message again, competing for access to the bus. The throughput model is given by:

$$Th_{can} = \left(1 - \frac{h_{size}}{F_{size}}\right)(1 - Loss_p)\left(\frac{F_{size}}{D_{can}}\right) \tag{10}$$

Delay Model

The delay experienced by CAN frames can be divided into the time taken for successful transmission, the time taken for a failed transmission (due to negative reception or reception of frames with errors), and the time that the transmitting node waits and listens caused by a busy channel or contention loss against a node transmitting a higher priority frame.

$$\begin{aligned} D_{can} = & \; successful\_transmission \\ & + unsuccessful\_transmission \\ & + medium\_inaccessible \end{aligned} \tag{11}$$

Where each model component is defined as:

$$\begin{aligned} successful\_transmission = & \left[(\psi_{idle} + \sum_{i=1}^{Id} \psi_{Cttn_i} \right. \\ & + \sum_{j=1}^{CTCA} \psi_{Tx_j} + \sum_{k=1}^{7} \psi_{IFS_k}) \\ & \left. (Id + CTCA + IFS)\tau \right] \end{aligned} \tag{12}$$

$$\begin{aligned} unsuccessful\_transmission = & [(\psi_{Idle} + \\ & \sum_{i=1}^{F_{size}} \sum_{j=1}^{i} \psi(\psi_{Cttn} + \psi_{Tx})_j \\ & \sum_{k=1}^{20} \psi_{Err_k})(F_{size} + 20)\tau] \end{aligned} \tag{13}$$

$$Medium\_Inaccessible = [(\psi_{Idle}(n\psi_{LW}))(nF_{size})\tau] \tag{14}$$

Thus, the CAN delay model is given by:

$$\begin{aligned} D_{can} = & \left[(\psi_{idle} + \sum_{i=1}^{Id} \psi_{Cttn_i} + \sum_{j=1}^{CTCA} \psi_{Tx_j} \right. \\ & \left. + \sum_{k=1}^{7} \psi_{IFS_k})(Id + CTCA + IFS)\tau \right] \\ & + \left[(\psi_{Idle} + \sum_{i=1}^{F_{size}} \sum_{j=1}^{i} \psi(\psi_{Cttn} + \psi_{Tx})_j \sum_{k=1}^{20} \psi_{Err_k}) \right. \\ & \left. (F_{size} + 20)\tau \right] + \left[(\psi_{Idle}(n\psi_{LW}))(nF_{size})\tau \right] \end{aligned} \tag{15}$$

CAN delay could be analyzed using other techniques; for example, the authors in [45] present a method for evaluating message response times for traffic traces and the transmission mode detection mechanism for CAN nodes to perform delay analysis.

4.2 LIN Protocol Delay and Throughput Model

The proposed analytical model for delay and throughput is based on what is described by the LIN specification package [46], where a LIN message frame time analysis on the bus is detailed. Some parameters of the LIN model are shown in Table 5.

Delay Model

There are different approaches to LIN time analysis. In [47], a model that calculates the response times of LIN frames in a LIN network implemented as a sub-bus in Volvo trucks

Table 5 Description of variables used for delay and throughput mathematical models of LIN protocol

Variable	Description
T_{Bit}	Bit time
$T_{Header\_Nominal}$	Time spent to transmit the header bits of the LIN frame by the Master
$T_{Response\_Nominal}$	Time spent to transmit the Response bits of the LIN frame by the slave
$T_{Frame\_Nominal}$	The sum of the time spent to transmit the header bits and the response bit that makes up a LIN frame
$T_{Frame\_Maximum}$	Time spent to transmit the header bits and the response bit that makes up a LIN frame plus 40% of additional time.
$T_{LIN\_Frame\_Slot}$	The time spent transmitting the header bits and the response bit that makes up a LIN frame plus 40% of additional time and a defined amount of time to compensate for jitter process.
T_{Jitter}	A defined amount of time to compensate for the inter-arrivals of LIN frames.
$T_{Ideal\_Frame\_Slot}$	Is equal to $T_{LIN\_Frame\_Slot}$
$T_{Real\_Frame\_Slot}$	Is the time of a LIN frame slot plus inter-frame guard time.
Mini slot	A defined amount of time that serves as a time base to transmit LIN frames (5 milliseconds is the default time used).
$T_{Inter\_Frame\_Space}$	Is an additional time added to an Ideal LIN frame slot to be expressed in mini-slots.
D_{tx}	The transmission delay, which is the serialization time of transmitting the first bit up to the last bit of the frame.
D_{prop}	Propagation delay
D_{pr}	The processing delay is the time the master and the slave node take to process the frame before it is transmitted and received.
D_q	Queuing delay
N_c	Network capacity

is presented. In another work [48], an algorithmic scheduling and packing approach is adopted to automate the process of generating a schedule that defines the specific time for each LIN frame that will be sent over the network to meet the synchronization requirements of the signals. A previous LIN network configuration is required to obtain an algorithmic optimization for the LIN frame scheduling. However, a detailed analysis with closed analytical formulas needs to be addressed. Furthermore, previous works did not address a general analysis of the LIN frame delay, considering transmission, queuing, processing, and propagation delays. The same applies to throughput analysis. In a different approach, the author of [49] employs a state machine and high-level computational tools to describe and implement the master and slave LIN nodes. The implementation was conducted in a microcontroller, and the response signals were analyzed. More recent work proposes a software and hardware tool for testing, error checking, and LIN time boundary evaluation. A computer running the software provides the parameters to the test and verification component, a hardware component that generates the LIN signals and communication [50]. Using this tool, it is possible to verify the performance of LIN under different error conditions and with different delay limits.

For the present work, the delay analysis is based on the time description presented in the [46] protocol specification. The delay experienced by LIN message frames sent over the LIN network consists of four components: propagation delay, transmission delay, queue delay, and processing delay. The LIN frame time analysis described in the specification considers the processing and transmission delay since once the frame header is scheduled

to be transmitted, it is processed and transmitted by the master node in the LIN network. The following equations provided by the LIN Specification Package 2.0 were analyzed and classified as transmission and processing delay:

$$T_{Header\_Nominal} = 34T_{Bit} \tag{16}$$

Where:

$$T_{Bit} = \frac{1}{Bitrate} \tag{17}$$

$$T_{Response\_Nominal} = 10(N_{databytes} + 1)T_{Bit} \tag{18}$$

$$\begin{aligned} T_{Frame\_Nominal} &= T_{Header\_Nominal} + T_{Response\_Nominal} \\ &= [10(N_{databytes}) + 44]T_{Bit} \end{aligned} \tag{19}$$

$$\begin{aligned} T_{Frame\_Maximum} &= 1.4T_{Frame\_Nominal} \\ &= 1.4[10(N_{databytes}) + 44]T_{Bit} \end{aligned} \tag{20}$$

$$\begin{aligned} T_{LIN\_Frame\_Slot} &= T_{Frame_{M}aximum} + T_{Jitter} \\ &= 1.4T_{Frame\_Nominal} + T_{Jitter} \\ &= T_{Ideal\_Frame\_Slot} \end{aligned} \tag{21}$$

$$T_{Inter\_Frame\_Space} = Minislot - Residue\left(\frac{T_{Ideal\_Frame\_Slot}}{Minislot}\right) \tag{22}$$

$$\begin{aligned} T_{Real\_Frame\_Slot} &= T_{Ideal\_Frame\_Slot} + T_{Inter\_Frame\_Space} \\ &= nT_{Time\_Base} > T_{Ideal\_Frame\_Slot} \end{aligned} \tag{23}$$

$$D_{tx} + D_{pr} = T_{Real\_Frame\_Slot} \tag{24}$$

Once the network begins to operate, the master node, through its master task, periodically processes the schedule in which the slots for individual messages are defined. Since LIN frames are transmitted periodically according to the program, the queuing technique is based on a deterministic round-robin scheme in which a send time is assigned to each frame. As a result, the queue delay during the LIN frame period, obtained directly by the LIN Schedule, is:

$$D_q = frame\_period\_based\_on\_Schedule(round - robin\ deterministic\_scheme) \tag{25}$$

In addition, it is important to consider the propagation delay, which is the time required for a bit to travel from the source node to the destination node. It is calculated by dividing the distance by the propagation speed (normally about $3x10^3$ m/s). The propagation delay is given by:

$$D_{prop} = \frac{wire\_length}{speed\_of\_light} \tag{26}$$

As a result, the latency model for LIN message frames transmitted in a LIN network is given by:

$$\begin{aligned} D_{Lin} &= D_{tx} + D_{prop} + D_{pr} + D_q \\ &= T_{Ream\_Frame\_Slot} + D_{prop} + D_q \end{aligned} \tag{27}$$

Throughput Model

The performance analysis for LIN network throughput was based on the bus utilization metric, a percentage of the network capacity (N_c) used at a specific time. An essential factor to consider when calculating bus utilization is the overhead of LIN frames, which for a frame containing an 8-byte payload, this overhead percentage is 63%. The average bus utilization of a LIN network is around 55% of the network capacity, which means that the overhead-to-payload ratio is high. Bus utilization is simply the time to transmit valuable data in a frame divided by the total time to send the entire frame on the network. As a result, the throughput model is given by:

$$Th_{Lin} = N_c \times Bus\_Utilization = N_c \times \frac{T_{Real\_Frame\_Slot}}{D_{Lin}} \tag{28}$$

Bridge Layer 2 Model

The interconnection of the network segments (LIN, CAN, and Ethernet) is analyzed. The network environment defined for this work consists of a LIN network segment formed by a master node and several slave nodes, a CAN network segment with several nodes, an Ethernet LAN network with several nodes, and the corresponding interconnection devices: CAN -LIN bi-directional bridge, bi-directional CAN-Ethernet bridge, and an Ethernet switch that connects to the CAN-Ethernet bridge as an Ethernet backbone connection (see Fig. 6).

The intra-vehicular network, based on the scenario shown in Fig. 6, presents a typical configuration and topology of the in-vehicle network as described and stated in [51–53]. The performance analysis of the CAN-LIN bridge, CAN-Ethernet bridge, and Ethernet switch in delay and throughput is done using queuing theory. The application of queue models to analyze the performance of bridges and switches simplifies the formulations of mathematical models for delay and throughput. Queuing models such as those presented in [34] define various performance metrics such as throughput and delay, representing how a bridge or switch works depending on the network protocols used for the network segments it interconnects.

CAN-LIN Bridge Model Specification

For the CAN-LIN bridge scenario presented in Fig. 7, the input interface of the bridge connected to the LIN domain and its frame processing scheme is modeled as an M/M/1/B queue. The input relating to the CAN domain is also modeled as an M/M/1/B queue, as shown in Fig. 7.

CAN-Ethernet Bridge Model Specification

The first scenario is the CAN-Ethernet bridge. According to Fig. 8, the input interface of the bridge connected to the CAN domain and its frame processing is modeled as an M/M/1/B queue. The output queue interface connecting Ethernet is also modeled as an M/M/1/B queue (see Fig. 8a) if the CAN payload within the Ethernet frame does not

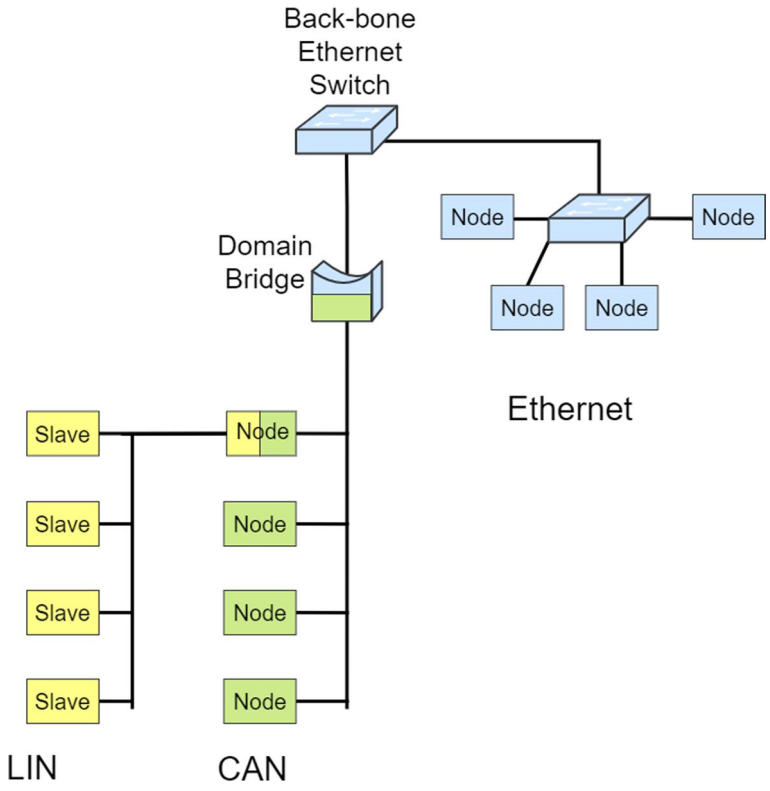


Fig. 6 Defined heterogeneous Intra-vehicular network composed of LIN, CAN, and Ethernet domains

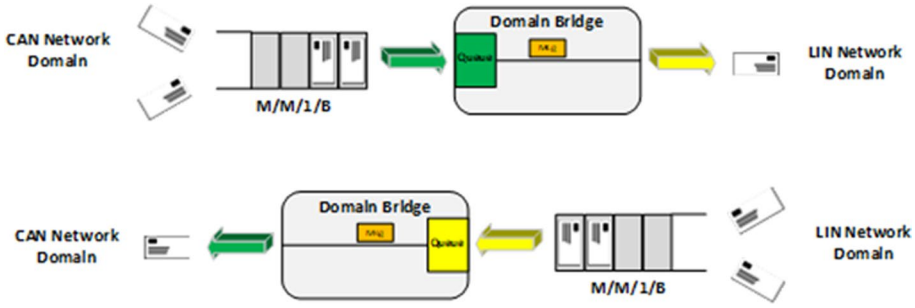


Fig. 7 Representation of the queuing system types applied to the CAN-LIN bridge

exceed 8 bytes, which is the maximum number of bytes that a CAN frame can store in its data/payload field.

On the other hand, the output queue interface connecting the Ethernet domain would perform frame fragmentation. As a result, the queue is modeled as an $M^m/M/1/B$ queue (see Fig. 8b). A representation of the network scenario for the frame fragmentation process is shown in Fig. 8b. The fragmentation rate is calculated by dividing the forwarding frame's payload size by the forward frame's payload size. This calculation

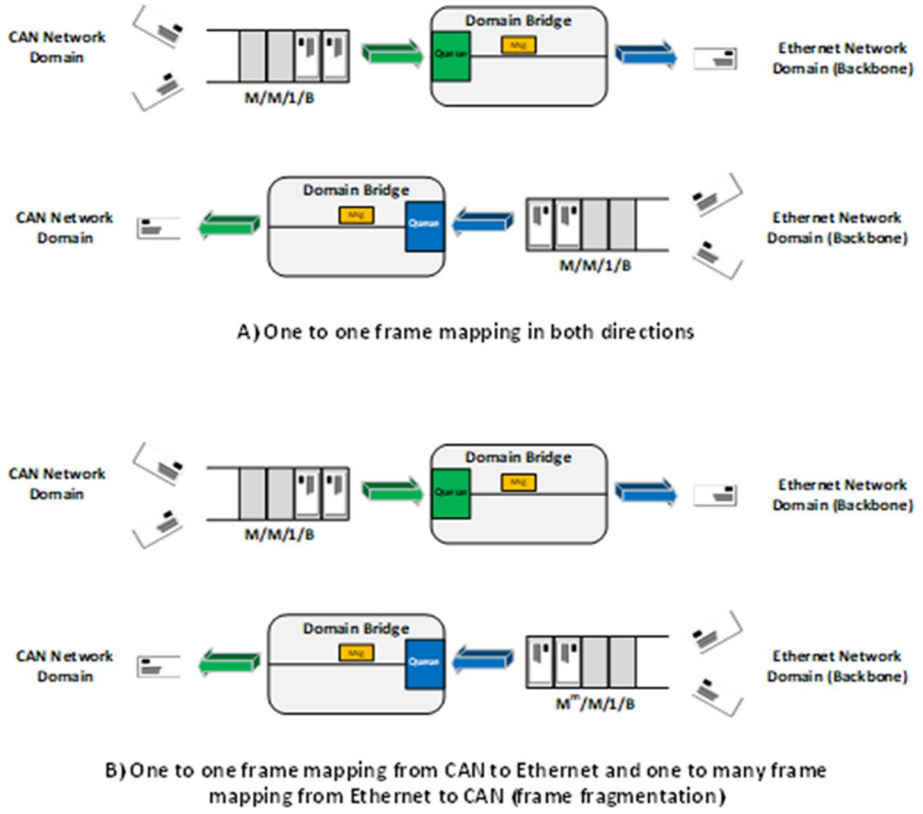


Fig. 8 Representation of the queuing system types applied to the CAN-Ethernet bridge

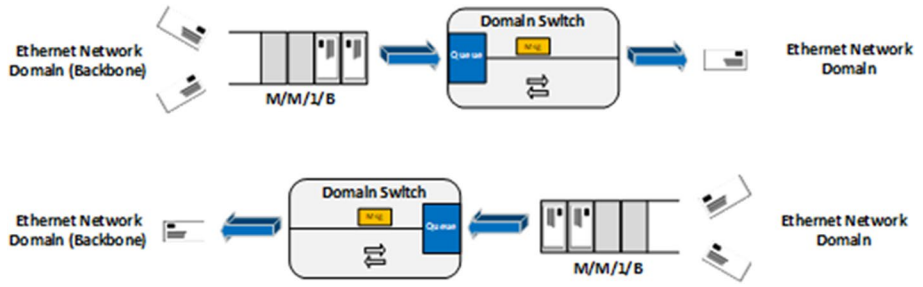


Fig. 9 Representation of the queuing system type applied to the Ethernet switch

establishes the number of fragments of the forwarded frame that will be encapsulated as separate frames.

Ethernet Switch Model Specification

For the switch scenario based on Fig. 9, the input interface of the switch connected to the Ethernet backbone and its frame processing scheme is modeled as an M/M/1/B queue.

The output queue interface connecting the Ethernet domain is also modeled as an M/M/1/B queue.

It is assumed that the behavior of the frames arriving at the bridge follows a Poisson distribution because it is the predominant model used to analyze traffic in voice and data networks. Another reason is that the vehicle network does not always experience bursts or as a common trait. Its behavior is a mixture of independent traffic processes and bursts of traffic. Concerning the Poisson distribution model assumed for this network scenario, depending on the value of the average frame arrival rate λ expressed in frames/sec divided by the analyzed network capacity N_c, the frame arrival probability value (a) is established according to the following formula:

$$a = \frac{\lambda}{N_c} \tag{29}$$

On the other hand, frame processing in which the frame is de-encapsulated, analyzed, and re-encapsulated in a format suitable for forwarding represents service times. The output process indicates These service times, which is assumed to have an exponential distribution. The following formula establishes the departure probability value (c) as a function of the average frame departure rate expressed μ in frames/sec divided by the network capacity N_c:

$$c = \frac{\mu}{N_c} \tag{30}$$

The queuing system model can be evaluated with the arrival and departure probabilities defined.

5 Heterogeneous Network Models

Considering the heterogeneous network defined for this work, the end-to-end one-way analysis for delay and throughput is based on the traffic flow from the source network domain that passes through intermediate bridges and other network domains (if that is the case) to the destination network domain. The one-way delay (e2e) model is formulated by adding the network delay (protocol-based media access delay) in all involved network domains and the processing and queuing delay experienced by frames on the bridge and switch across the transmission path. Unlike the one-way delay model (e2e), the one-way throughput (e2e) is formulated by taking the minimum observed throughput from the source network domain that passes through the intermediate bridge (s) and other network domains (if that is the case) to the destination network domain. End-to-end throughput is also modeled based on the transmission path flow. Table 6 contains the delay and throughput of the end-to-end unidirectional models based on the different transmission flow configurations.

Each delay and throughput model component for different flows is defined in Table 7.

Table 6 One-way end-to-end delay and throughput models for the LIN, CAN, and Ethernet heterogeneous intra-vehicular network

Traffic Flow: end-to-end Delay and Throughput Models					
Path Label	Tx Node	Rx Node	Bridge	Delay	Throughput
1a	LIN	CAN	Br_{L_C}	$Delay_{e2e} = D_{LIN} + D_{BR_{L\_C}} + D_{CAN}$	$Th_{e2e} = min(Th_{LIN}, Th_{Br_{L\_C}}, Th_{CAN})$
1b	CAN	LIN	Br_{L_C}	$Delay_{e2e} = Delay_{CAN} + Delay_{Br_{L\_C}} + D_{LIN}$	$Th_{e2e} = min(Th_{CAN}, Th_{Br_{L\_C}}, Th_{LIN})$
2a	CAN	Eth	$Br_{C\_EB}\, SW_{EB\_E}$	$Delay_{e2e} = D_{CAN} + D_{Br_{C\_EB}} + D_{SW_{EB\_E}} + D_{Eth}$	$Th_{e2e} = min(Th_{CAN}, Th_{Br_{C\_EB}}, Th_{SW_{EB\_E}}, Th_{Eth})$
2b	Eth	CAN	$SW_{EB\_E}\, Br_{C\_EB}$	$Delay_{e2e} = D_{Eth} + D_{SW_{EB\_E}} + D_{Br_{C\_EB}} + D_{CAN}$	$Th_{e2e} = min(Th_{Eth}, Th_{SW_{EB\_E}}, Th_{Br_{C\_EB}}, Th_{CAN})$
3a	LIN	Eth	$Br_{L\_C}\, Br_{C\_EB}\, SWEB\_E$	$Delaye2e = D_{LIN} + D_{Br_{L\_C}} + D_{CAN} + D_{Br_{C\_EB}} + D_{SW_{EB\_E}} + D_{Eth}$	$Th_{e2e} = min(Th_{LIN}, Th_{Br_{L\_C}}, Th_{CAN}, Th_{Br_{C\_EB}}, Th_{SW_{EB\_E}}, Th_{Eth})$
3b	Eth	LIN	$SWEB\_E\, Br_{C\_EB}\, Br_{L\_C}$	$Delaye2e = D_{Eth} + D_{SW_{EB\_E}} + D_{Br_{C\_EB}} + D_{CAN} + D_{Br_{L\_C}} + D_{LIN}$	$Th_{e2e} = min(Th_{Eth}, Th_{SW_{EB\_E}}, Th_{Br_{C\_EB}}, Th_{CAN}, Th_{Br_{L\_C}}, Th_{LIN})$

Table 7 Description of variables used in the delay and throughput one-way end-to-end models

Variable	Description
$Delay_{e2e}$	Stands for one-way end-to-end delay
Th_{e2e}	Stands for one-way end-to-end throughput
D_{Eth}	Stands for Ethernet network domain delay
$D_{SW_{EB\_E}}$	Stands for Ethernet switch delay. The sub-indices stand for Ethernet Backbone to Ethernet (EB_E)
$D_{Br_{C\_EB}}$	Stands for CAN-Ethernet bridge delay. The sub-indices stand for CAN to Ethernet Backbone (C_EB)
D_{Can}	Stands for CAN network domain delay
$D_{Br_{L\_C}}$	Stands for LIN-CAN bridge delay. The sub-indices stand for LIN to CAN (L_C)
D_{Lin}	Stands for LIN network domain delay
Th_{Eth}	Stands for Ethernet network domain throughput
$Th_{SW_{EB\_E}}$	Stands for Ethernet switch throughput. The sub-indices stand for Ethernet Backbone to Ethernet (EB_E)
$Th_{Br_{C\_EB}}$	Stands for CAN-Ethernet bridge throughput. The sub-indices stand for CAN to Ethernet Backbone (C_EB)
Th_{Can}	Stands for CAN network domain throughput
$Th_{Br_{L\_C}}$	Stands for LIN-CAN bridge throughput. The sub-indices stand for LIN to CAN (L_C)
Th_{Lin}	Stands for LIN network domain throughput

6 Results

Simulations with Vector CANoe Demo Version 10 validate the CAN, LIN, and end-to-end network traffic flow throughput and delay models. Based on [54], the different gateways were also developed using tools provided by CANoe. It is important to note that the performance analysis for delay and throughput is based on a general network communication scenario where the frames/messages are generated at the network nodes and transmitted according to the network protocol that controls them. The network created for this project does not focus on vehicle application scenarios (e.g., vehicle door functions).

The following are the objectives of the Vector CANoe simulation based on the project study case:

1. Simulation of the following individual/homogeneous networks: LIN, CAN, and Ethernet network as individual scenarios.
2. Heterogeneous network simulation consisting of a LIN, CAN, and Ethernet domain, an Ethernet backbone link, a CAN-LIN bi-directional bridge, a CAN-Ethernet gateway, and an Ethernet switch.

This section discusses the simulation results obtained for the homogeneous network delay and throughput model validation cases, as Table 8 specified.

6.1 LIN Delay and Throughput Model Validation

Figures 10 and 11 show the analytical models and simulation delay and throughput results for the LIN variable payload scenario. On the other hand, Figs. 12 and 13 show the same metrics for the variable node number scenario.

The results presented in Figs. 10 and 11 represent a scenario in which a single LIN frame is transmitted on the network. This frame payload ranges from 1 to 8 (maximum size) payload. On the other hand, Figs. 12 and 13 show the results of a scenario in which three slave nodes are scheduled to transmit from 1 to 8 maximum size frame. The results (model and simulation) are very similar in both cases. Subsequently, this is analytically validated using correlation measures and RMSE (mean square error).

Table 8 One-way end-to-end delay and throughput models for the LIN, CAN, and Ethernet heterogeneous intra-vehicular network

Single Protocol Delay and Throughput Model Validation and Performance Analysis				
	Transmission Case		Variation	
Protocol	Single Frame	Multiples Frame	Frame Size	Number of Frames
LIN	✓	✓	1–8 bytes payload	1–8 frame of 8 bytes payload
CAN	✓	✓	1–8 bytes payload	1–8 frame of 8 bytes payload
Ethernet	✓	✓	payload size (46, 64, 128, 256, 512, 768, 1024, 1500 bytes)	1–8 frame of 46 bytes payload

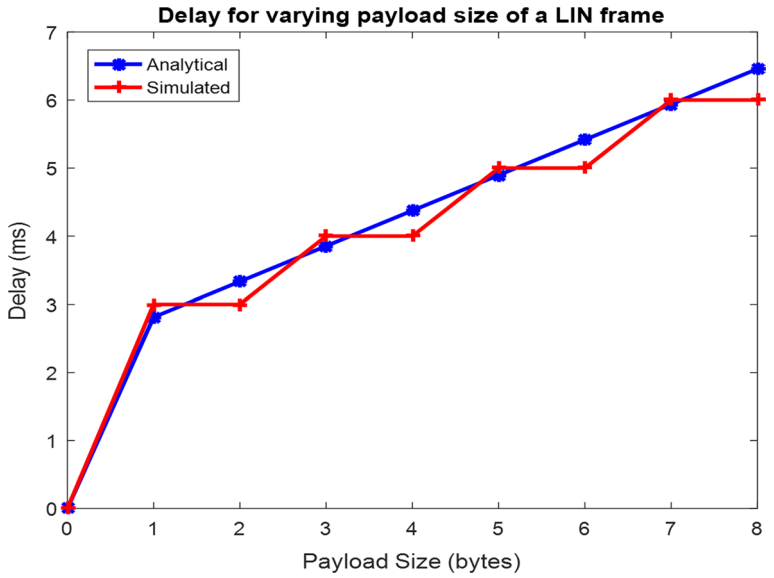


Fig. 10 Analytical (blue) and simulation (red) results for Delay when varying payload size of a single LIN frame. (Color figure online)

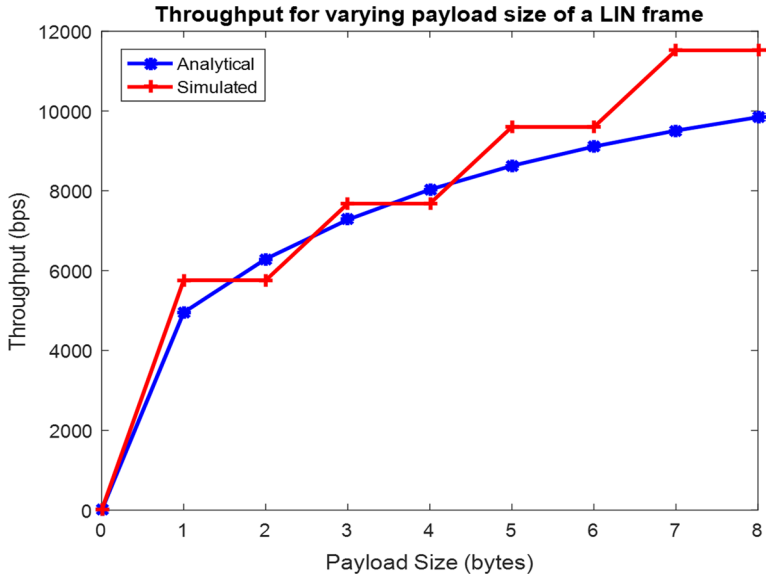


Fig. 11 Analytical (blue) and simulation (red) results for Throughput when varying payload size of a single LIN frame. (Color figure online)

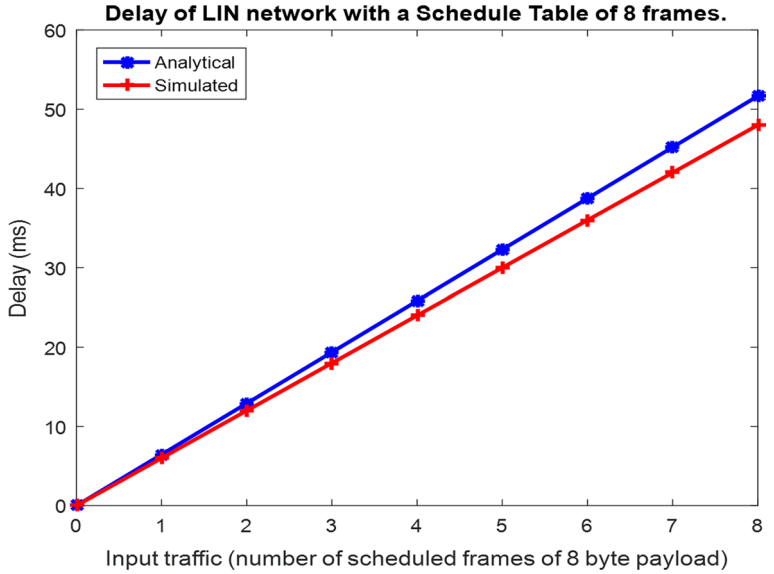


Fig. 12 Analytical (blue) and simulation (red) results for Delay when varying frames number. (Color figure online)

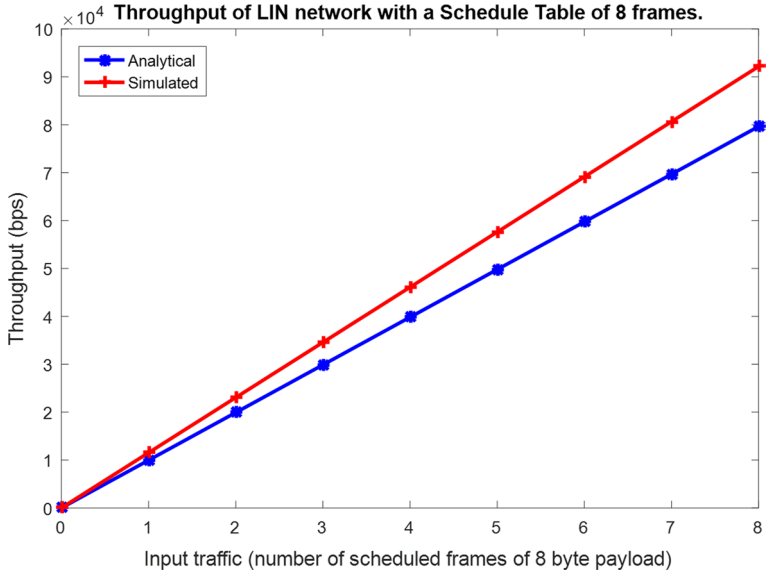


Fig. 13 Analytical (blue) and simulation (red) results for Throughput when varying frames number. (Color figure online)

6.2 CAN Delay and Throughput Model Validation

Figures 14 and 15 show the analytical models and the simulation delay and throughput

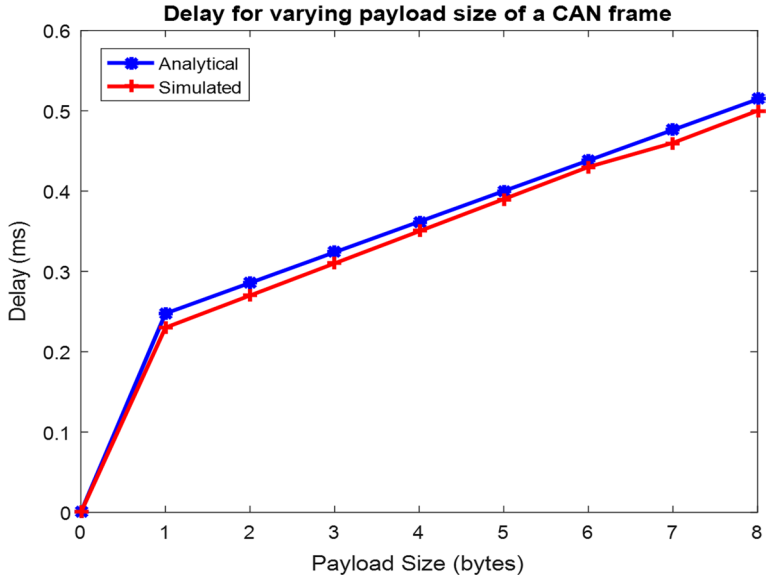


Fig. 14 Analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of a single frame sent from the transmitting node to the receiving node in a CAN network. (Color figure online)

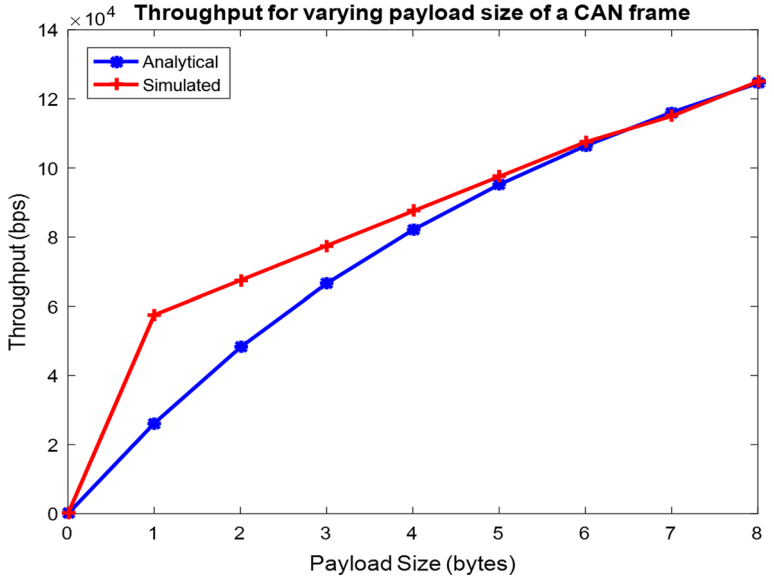


Fig. 15 Analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of a single frame sent from the transmitting node to the receiving node in a CAN network. (Color figure online)

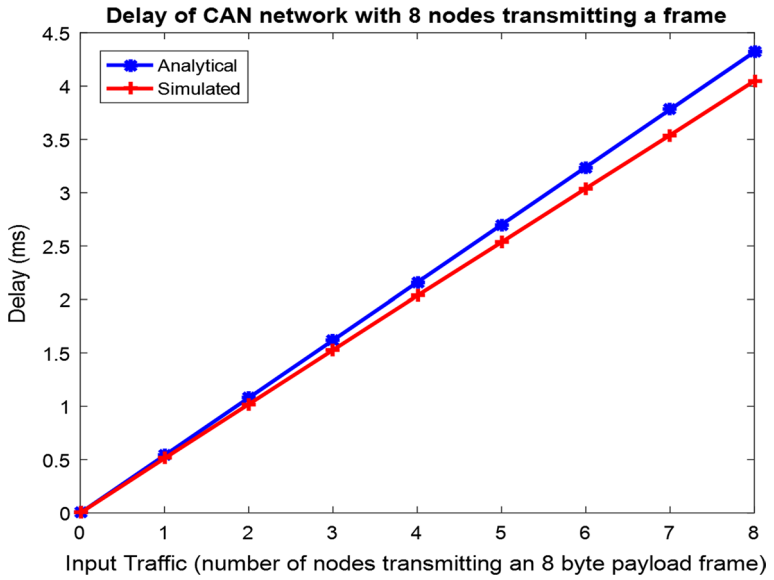


Fig. 16 Analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of frames sent in a CAN network. (Color figure online)

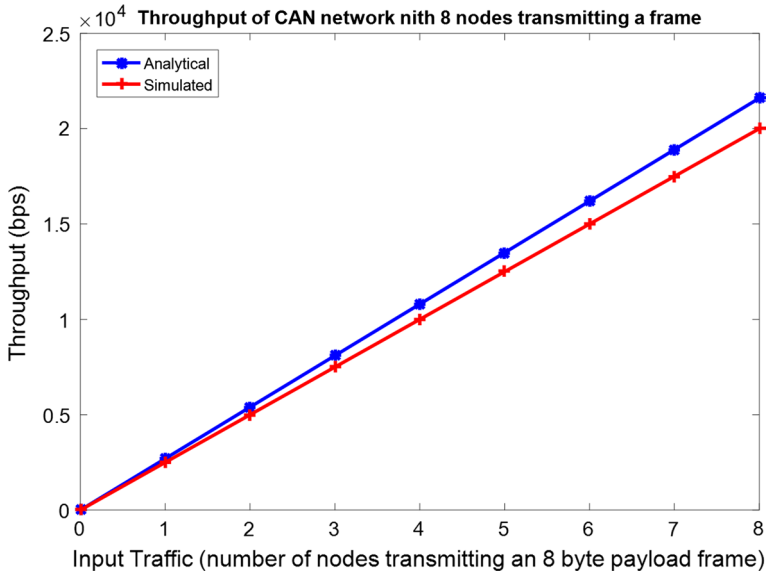


Fig. 17 Analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of frames sent in a CAN network. (Color figure online)

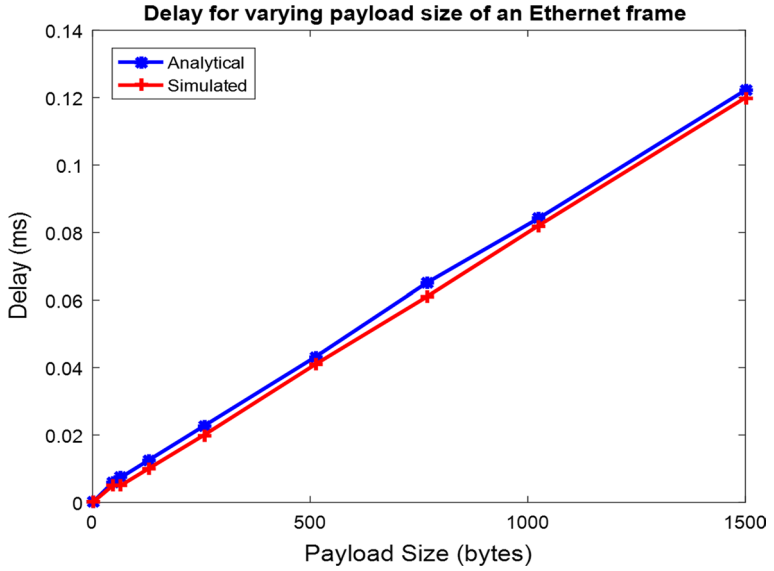


Fig. 18 Analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of a single frame sent from the transmitting node to the receiving node in an Ethernet bus network. (Color figure online)

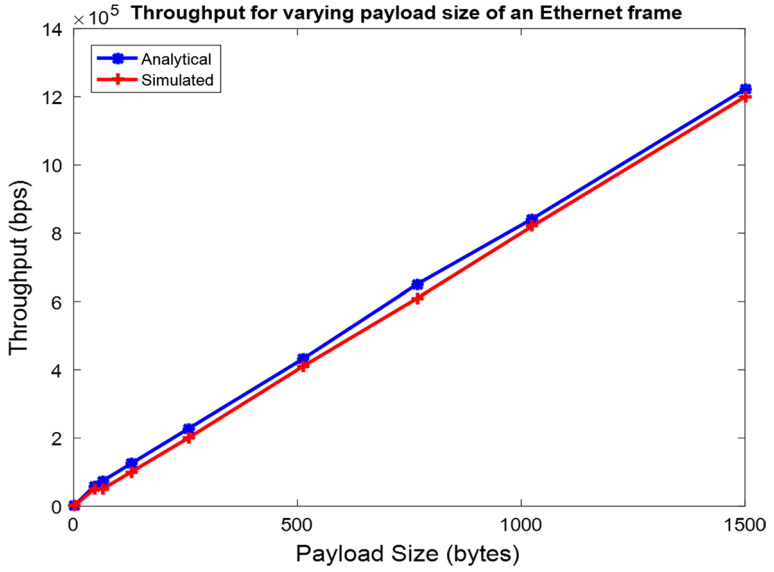


Fig. 19 Analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of a single frame sent from the transmitting node to the receiving node in an Ethernet bus network. (Color figure online)

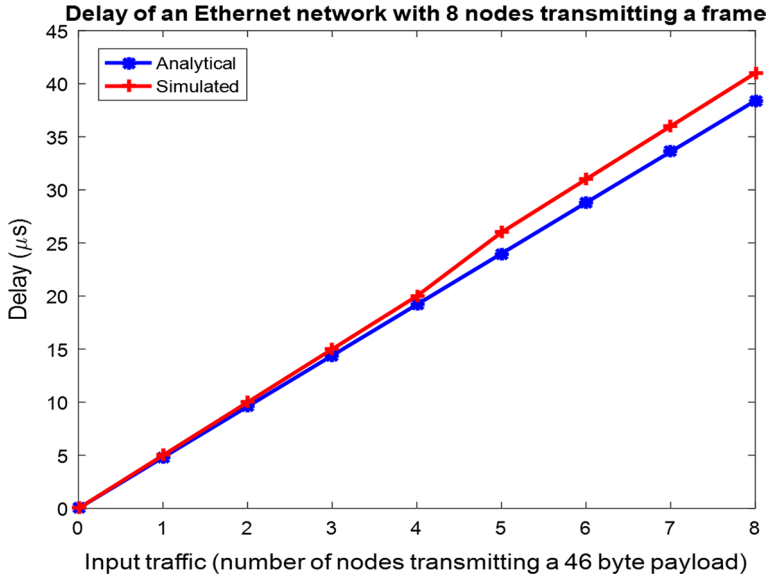


Fig. 20 Analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of frames sent in an Ethernet bus network. (Color figure online)

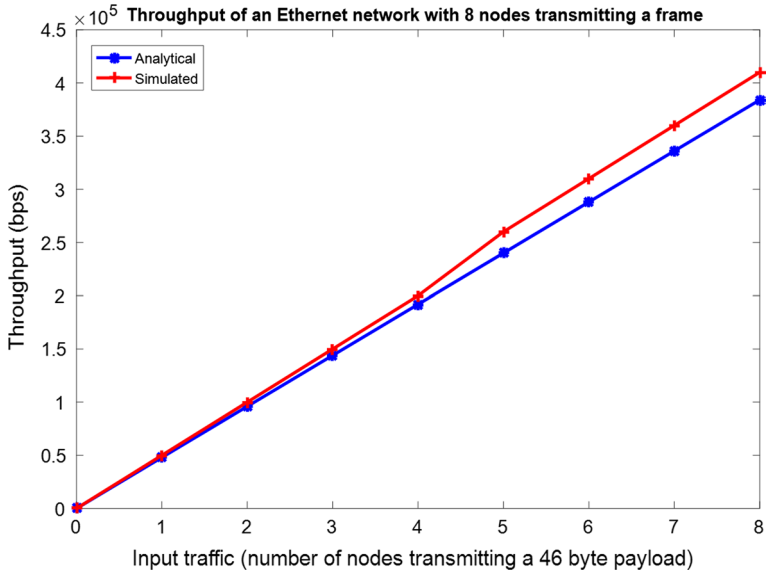


Fig. 21 Analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of frames sent in an Ethernet bus network. (Color figure online)

Table 9 Frame transmission configurations for heterogeneous network scenarios to evaluate one-way end-to-end delay and throughput performance

Single Protocol Delay and Throughput Model Validation and Performance Analysis								
	Tranmission case		Variation		Validation			
					Correlation		Root Mean Square Error (RMSE)	
Protocol	Single Frame	Multiple Frame	Frame Size	Number of Frames	Delay(s)	Th (bps)	Delay (s)	Th (bps)
LIN	✓		1–8 bytes payload		0.9911	0.9911	0.00028	1014
LIN		✓		1–8 frames of 8 bytes payload	0.9999	0.9999	0.0022	7457
CAN	✓		1–8 bytes payload		0.9996	0.968	0.000047	12,955
CAN		✓		1–8 frames of 8 bytes payload	0.9999	0.9999	0.000158	952
Ethernet	✓		payload size (46, 64, 128, 256, 512, 768, 1021, 1500 bytes)		0.9997	0.9997	0.000002	23,912
Ethernet		✓		1–8 frames of 46 bytes payload	0.9998	0.9998	0.000001	13,832

results for the variable payload (1 to 8 bytes). Figures 16 and 17 show delay and throughput results for a variable number of nodes (four nodes transmitting 1 to 8 8-byte payload frames). In both cases, the nodes transmit at 5 Kbps.

6.3 CSMA/CD Delay and Throughput Model Validation

Figures 18 and 19 show the variable payload scenario's analytical models and simulation delay and throughput results. The same metric results for a varying number of nodes on an Ethernet network are shown in Figs. 20 and 21 for delay and throughput, respectively.

The first scenario (Figs. 18 and 19) is composed of two nodes connected to an Ethernet bus. One of these nodes transmits one frame with a payload varying in {46,64,128,256,512,768,1024,1500} bytes. For Figs. 20 and 21, the Ethernet network consists of 4 nodes transmitting eight minimum payload frame sizes (46 bytes). It is observed, in Figs. 18, 19, 20 and 21, that the model estimations and the simulation results are very similar.

Table 9 provides a concise report of the results regarding validating homogeneous network delay and throughput models by correlation and the RMSE evaluation.

Referring to Table 9, the correlation values obtained for the different cases are very similar, the lowest correlation value being 0.9688 and the highest 0.9999. This implies that the results of the delay and throughput analytical models compared to the simulation ones are almost identical. Meanwhile, there is an insignificant difference between the analytical LIN delay model and the simulated LIN delay results for the mean square error values obtained. Regarding the transmission cases: (i) for single frames, the RMSE was 0.28 ms, and (ii) for multiple frames, it was 2.2 ms. As a result, the estimated simulation delay closely matches the delay obtained from the analytical model. The same can be said for CAN and Ethernet transmission cases. For the CAN analytical delay model and delay simulation results, RMSE evaluation returned an error of 47 microseconds μs for the single-frame case and an error of 158 μs for the multiple-frame case. The Ethernet analytical delay model and simulation delay results evaluation through RMSE show the slightest error, 2.4 μs for the case of a single frame and 1.6 μs for the multiple-frames case.

On the other hand, the mean square error values obtained for the LIN analytical throughput model and the LIN throughput simulation results concerning both transmission

Table 10 Frame transmission configurations for heterogeneous network scenarios to evaluate one-way end-to-end delay and throughput performance

Transmission configurations for end-to-end performance analysis					
Path Label	Tx Node	Rx Node	Varying payload size	Varying number of transmitting/receiving nodes	Single Frame/ Multiple Frames/ Frame Fragmentation
1a	LIN	CAN	✓	✓	Single and Multiple
1b	CAN	LIN	✓	✓	Single and Multiple
2a	CAN	Ethernet		✓	Single and Multiple
2b	Ethernet	CAN	✓	✓	Single,Multiple, Fragmentation
3a	LIN	Ethernet	✓	✓	Single and Multiple
3b	Ethernet	LIN	✓	✓	Single,Multiple, Fragmentation

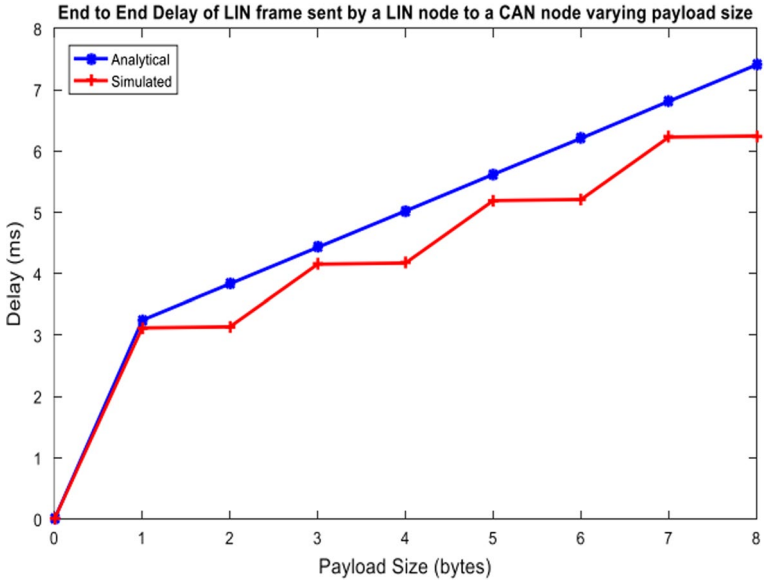


Fig. 22 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of a frame sent from a LIN node to a CAN node. (Color figure online)

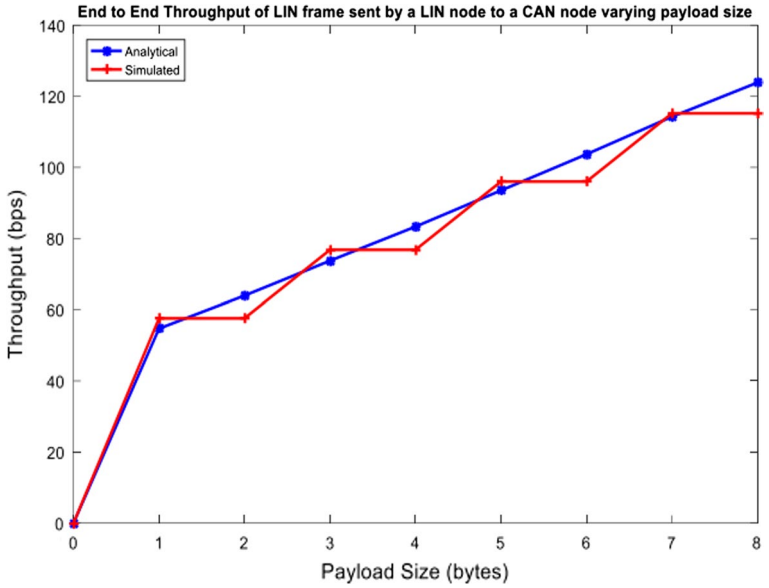


Fig. 23 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) throughput when varying payload size of a frame sent from a LIN node to a CAN node. (Color figure online)

cases were less favorable compared to the RMSE values of the delay but still were acceptable. For the single-frame case, the RMSE was 1014 bps (1014 Kbps); for the

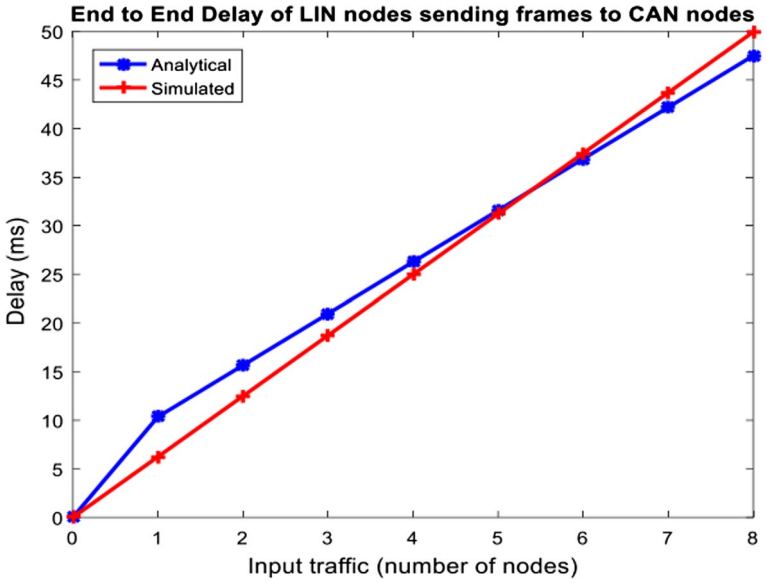


Fig. 24 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of LIN nodes sending an 8-byte payload frame to the CAN network segment. (Color figure online)

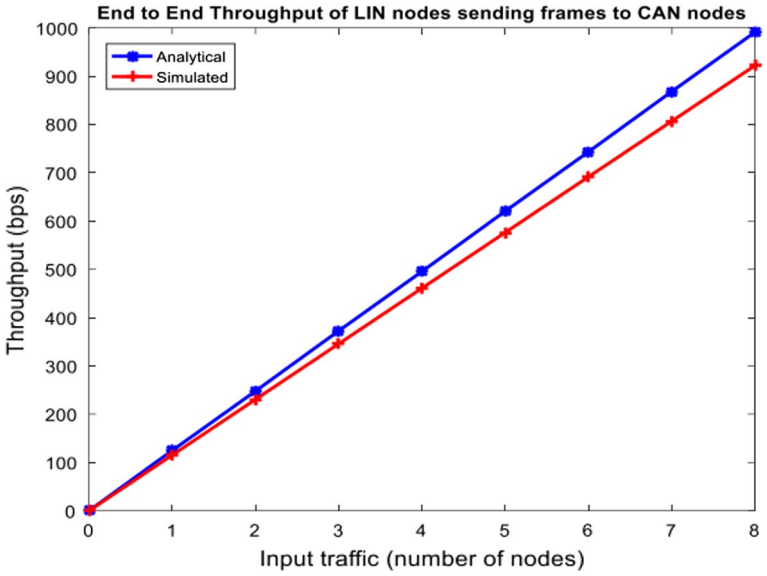


Fig. 25 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of LIN nodes sending an 8-byte payload frame to the CAN network segment. (Color figure online)

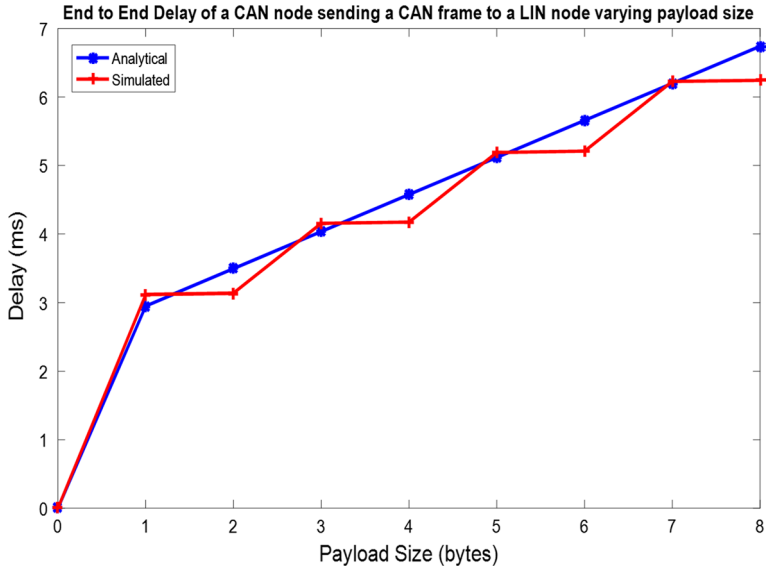


Fig. 26 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of a frame sent from a CAN node to a LIN node. (Color figure online)

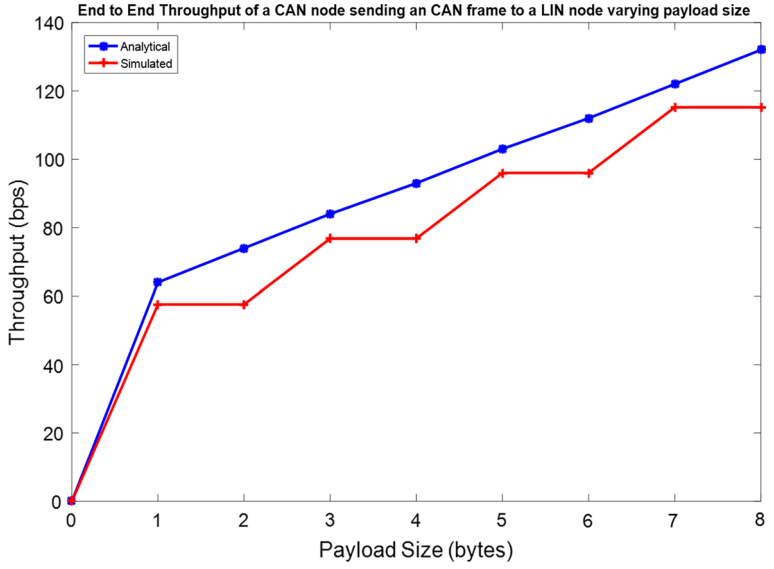


Fig. 27 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of a frame sent from a CAN node to a LIN node. (Color figure online)

multi-frame case, it was 7457 bps (7,457 Kbps). Furthermore, when comparing the measured values of the CAN analytical throughput model and the results of the CAN

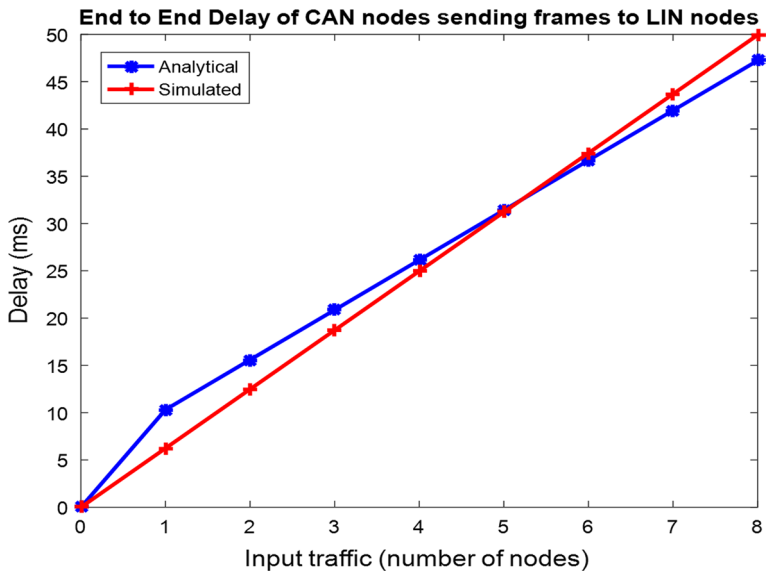


Fig. 28 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of CAN nodes sending an 8-byte payload frame to the LIN network segment. (Color figure online)

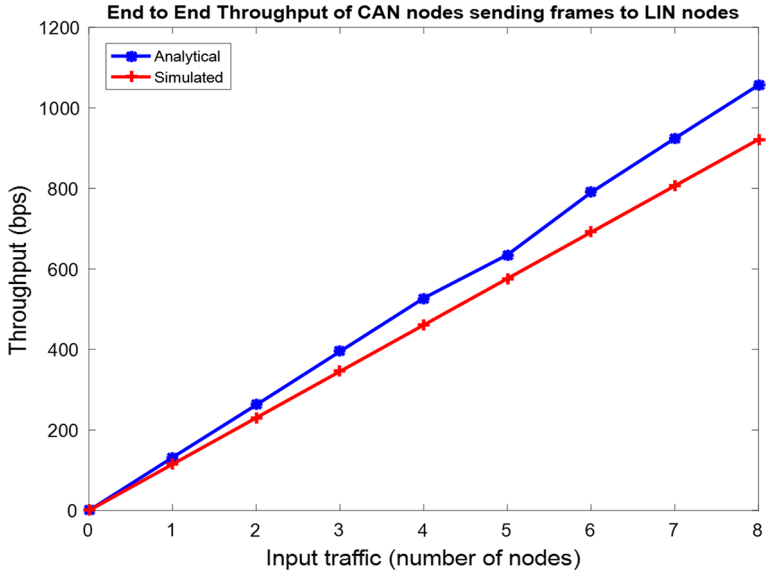


Fig. 29 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of CAN nodes sending an 8-byte payload frame to the LIN network segment. (Color figure online)

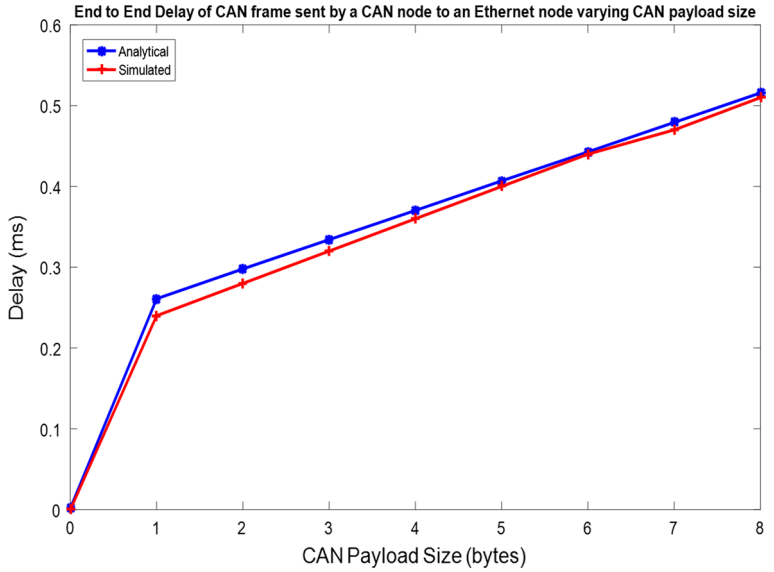


Fig. 30 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying CAN payload size of a frame sent from a CAN node to an Ethernet node. (Color figure online)

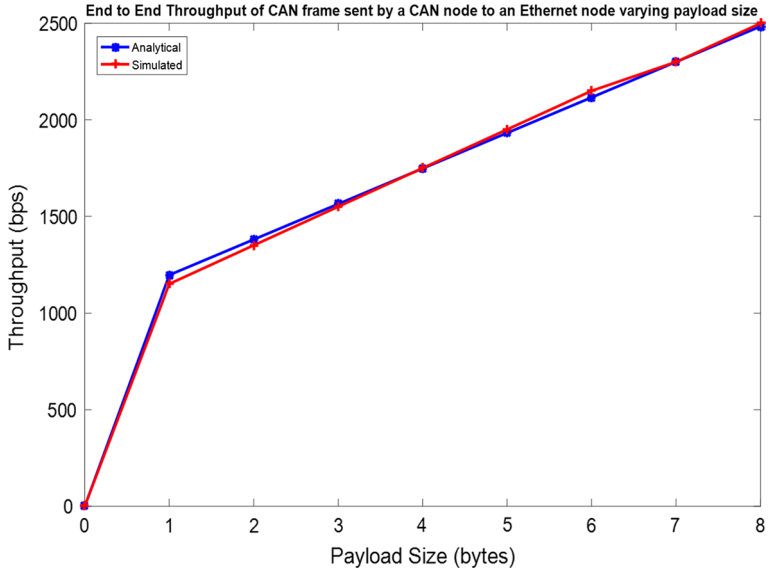


Fig. 31 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying CAN payload size of a frame sent from a CAN node to an Ethernet node. (Color figure online)

throughput simulation in both transmission cases, the obtained RMSE evaluation result indicated negligible error. For the single frame case, the RMSE was 12,955 bps (12.955

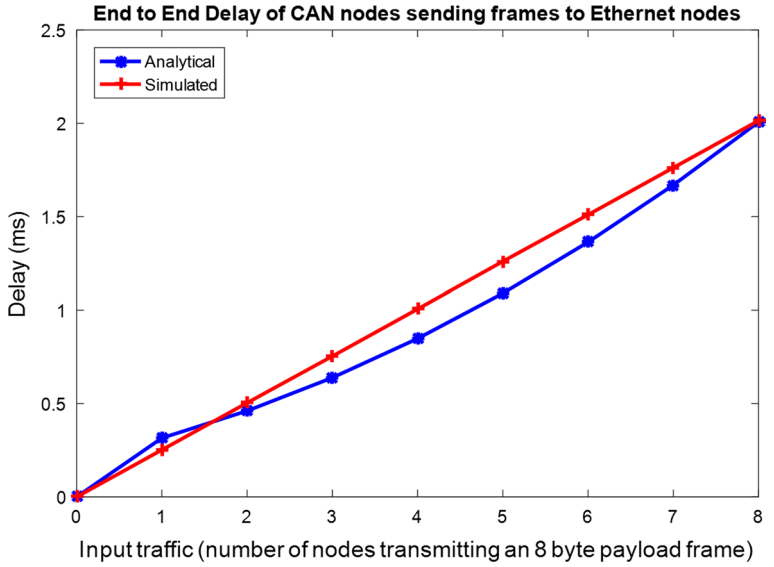


Fig. 32 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of CAN nodes sending an 8-byte payload frame to the Ethernet network segment. (Color figure online)

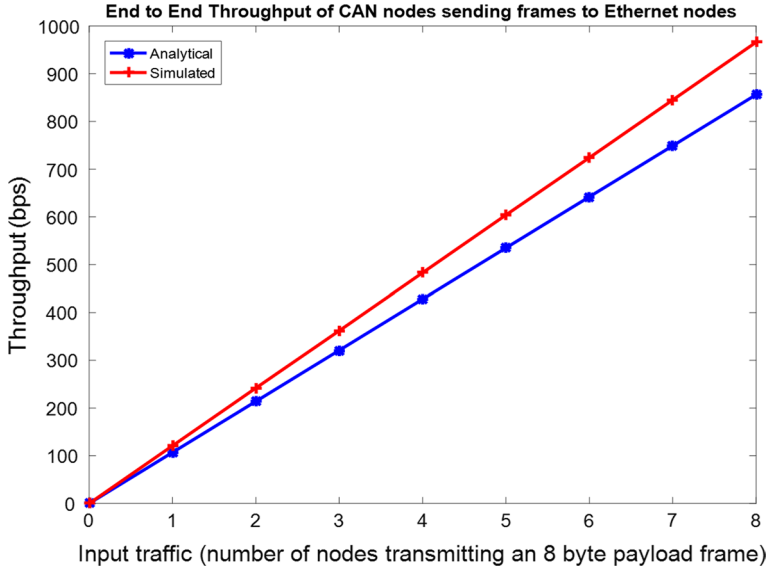


Fig. 33 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of CAN nodes sending an 8-byte payload frame to the Ethernet network segment. (Color figure online)

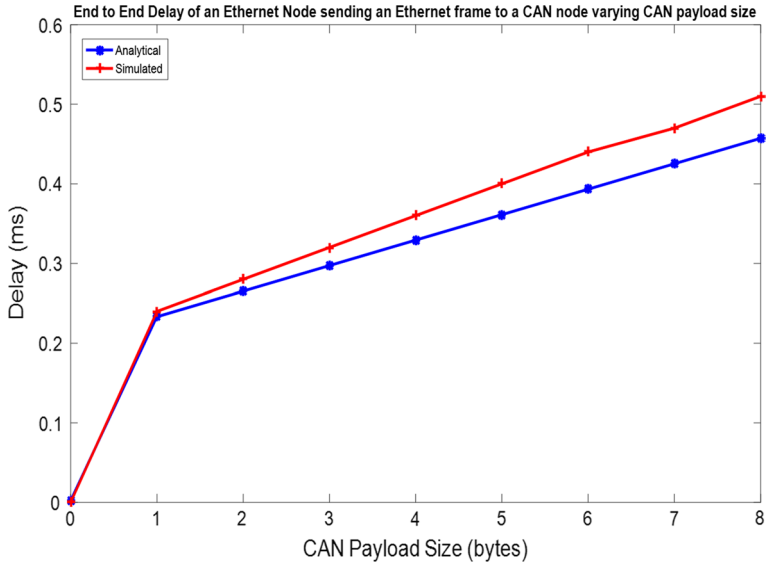


Fig. 34 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of CAN data inside a 46-byte payload Ethernet frame sent from an Ethernet node to a CAN node. (Color figure online)

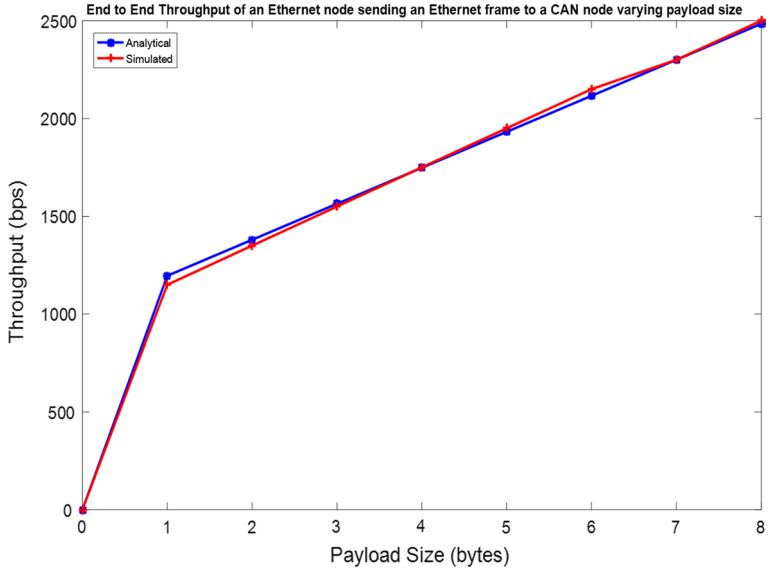


Fig. 35 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of CAN data inside a 46-byte payload Ethernet frame sent from an Ethernet node to a CAN node. (Color figure online)

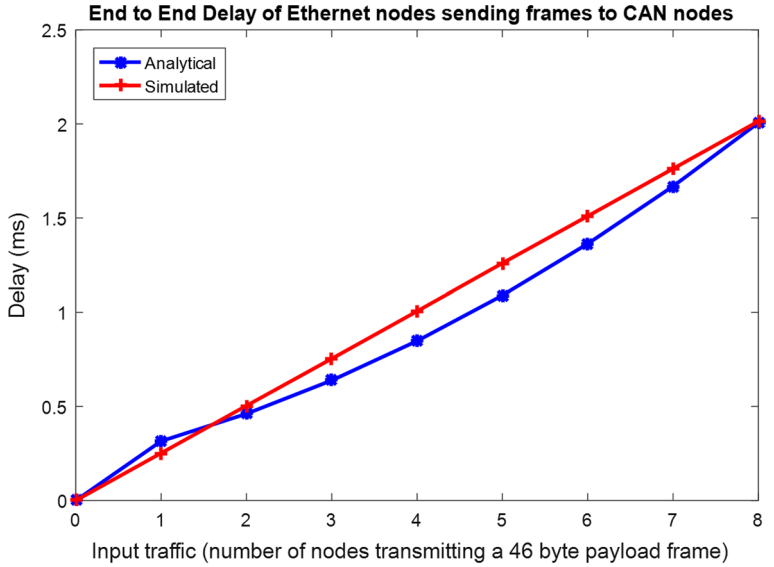


Fig. 36 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of Ethernet nodes sending a 46-byte payload frame containing 8 bytes of CAN data to the CAN network segment. (Color figure online)

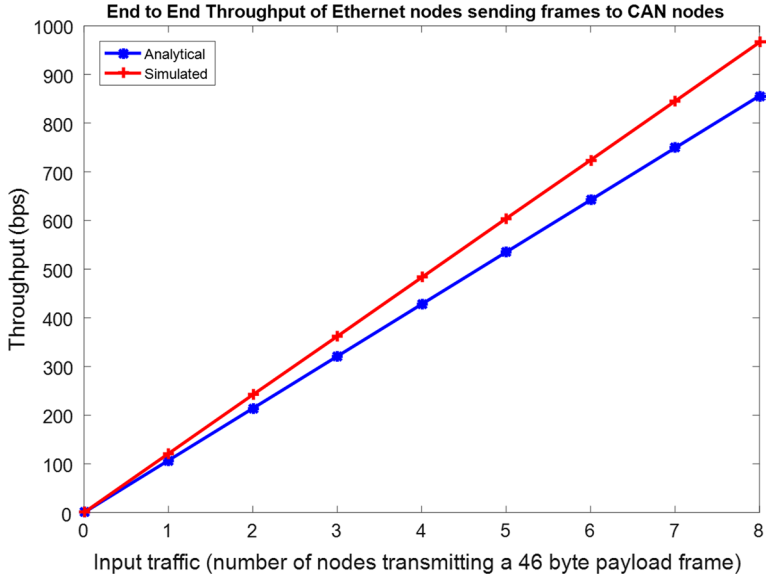


Fig. 37 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of Ethernet nodes sending a 46-byte payload frame containing 8 bytes of CAN data to the CAN network segment. (Color figure online)

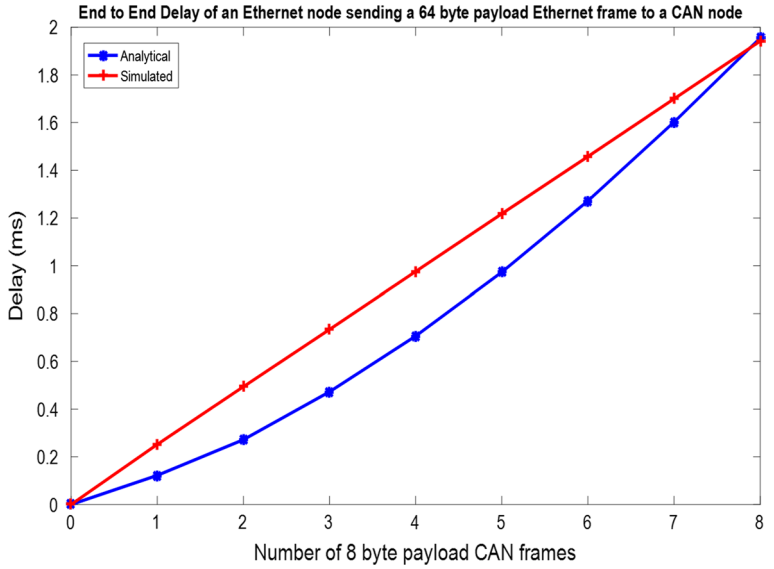


Fig. 38 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay and throughput when varying payload size (in steps of 8 bytes) of CAN data inside a 64-byte payload Ethernet frame sent from an Ethernet node to CAN node. (Color figure online)

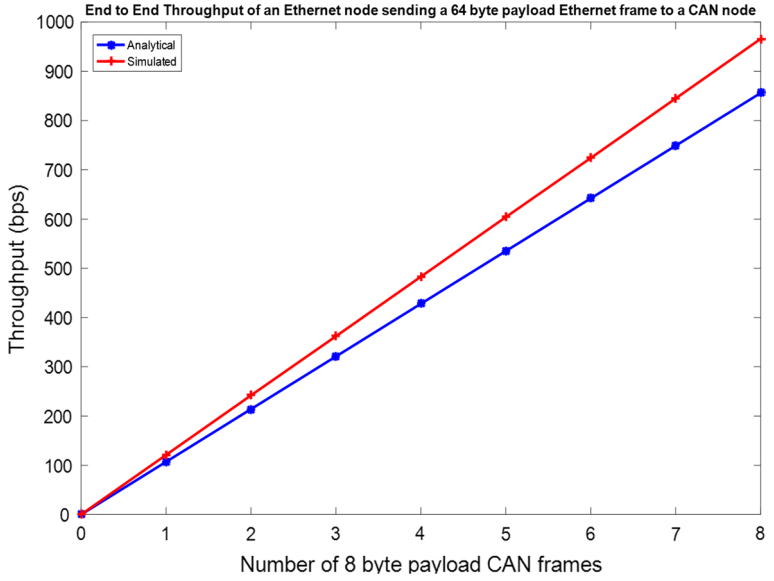


Fig. 39 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay and throughput when varying payload size (in steps of 8 bytes) of CAN data inside a 64-byte payload Ethernet frame sent from an Ethernet node to CAN node. (Color figure online)

Kbps); for the multiple frame case, it was 952 bps (0.952 Kbps). The same can be said for Ethernet transmission cases. The evaluation of the analytical throughput model for Ethernet and the results obtained from the simulation for the RMSE gave 23,912 bps for the single frame case and 15,832 bps for the multiple frame case.

As a result, the simulated throughput values for the homogeneous LIN, CAN, and Ethernet networks are a close fit and an excellent approximation to the results obtained from the analytical throughput model.

6.4 End-to-End Heterogeneous Network Model Validation

End-to-end delay and throughput are evaluated, defined by six different traffic flow paths, according to Table 10. For those six traffic flow paths, configurations such as the variable payload size, the number of transmitting and receiving nodes, and the type of frame encapsulation, single frame encapsulation, or frame fragmentation, are considered.

6.4.1 LIN-CAN End-to-End Performance Analysis

Figures 22 and 23 show the analytical model and simulation delay and throughput results for the variable payload of path 1a. The results for the same path, with a variable number of nodes, are shown in Fig. 24 for the delay and Fig. 25 for throughput.

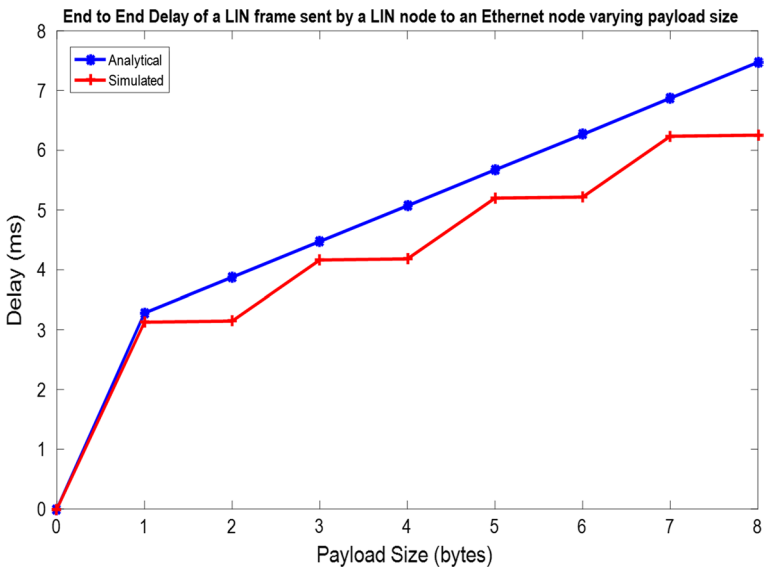


Fig. 40 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of a frame sent from a LIN node to an Ethernet node. (Color figure online)

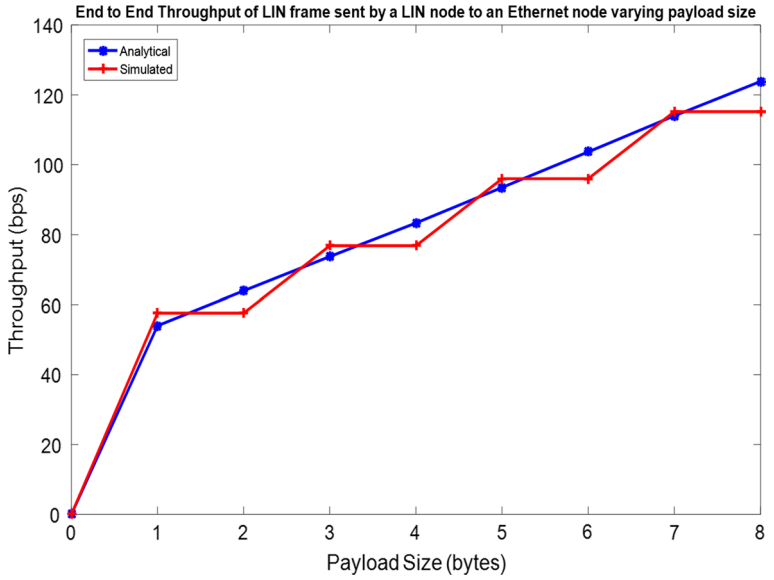


Fig. 41 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of a frame sent from a LIN node to an Ethernet node. (Color figure online)

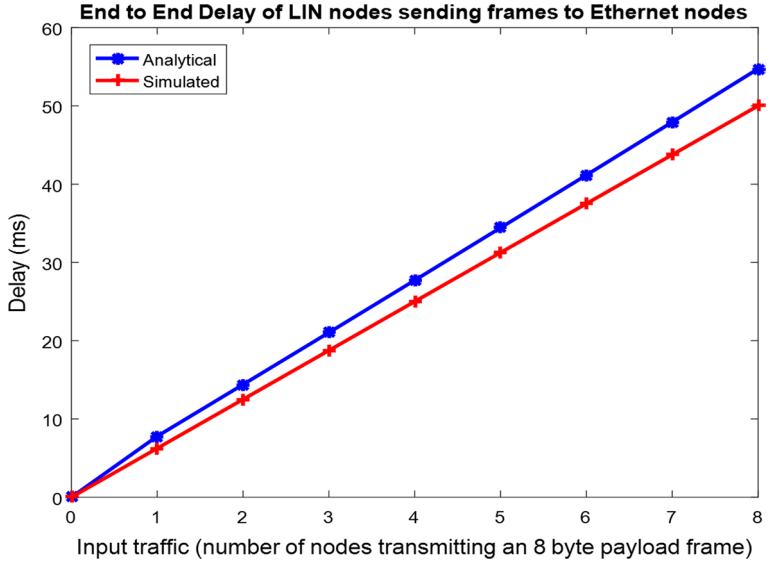


Fig. 42 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of LIN nodes sending an 8-byte payload frame to the Ethernet network segment. (Color figure online)

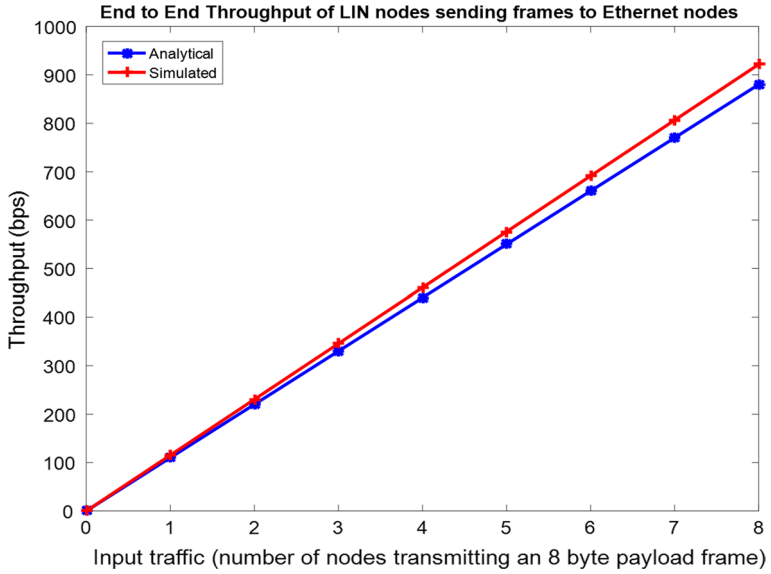


Fig. 43 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of LIN nodes sending an 8-byte payload frame to the Ethernet network segment. (Color figure online)

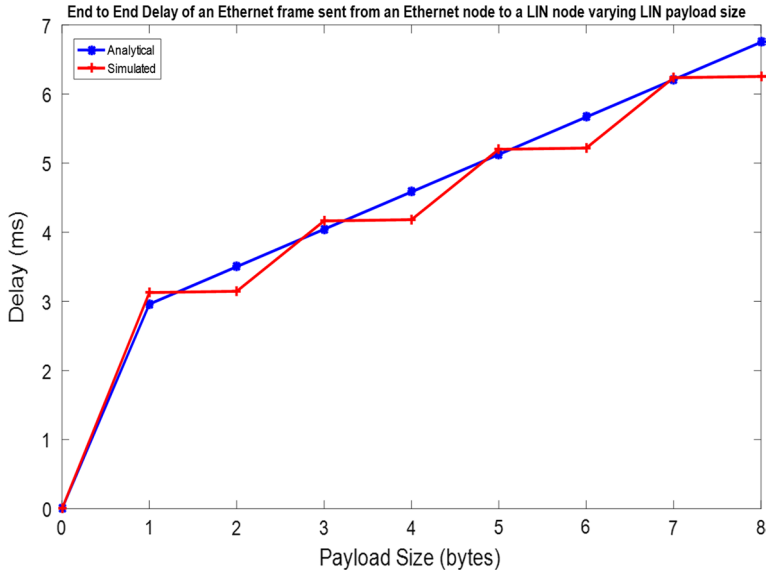


Fig. 44 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size of LIN data inside a 46-byte payload Ethernet frame sent from an Ethernet node to a LIN node. (Color figure online)

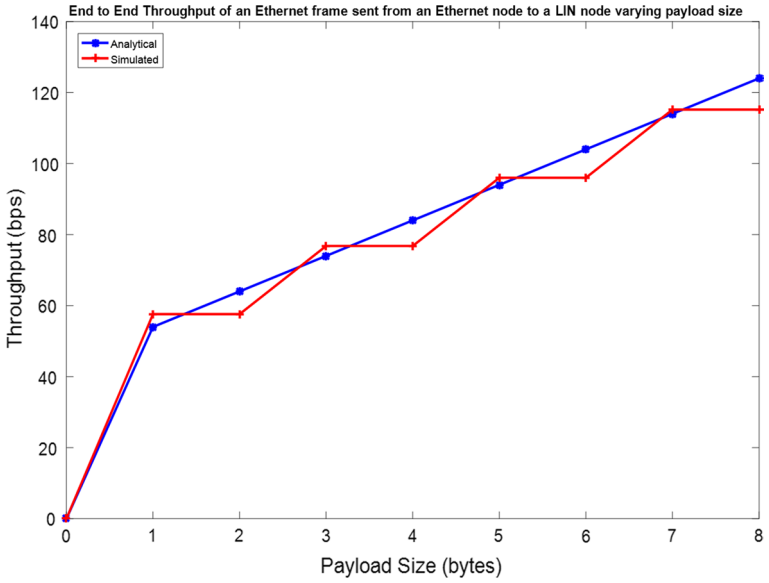


Fig. 45 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size of LIN data inside a 46-byte payload Ethernet frame sent from an Ethernet node to a LIN node. (Color figure online)

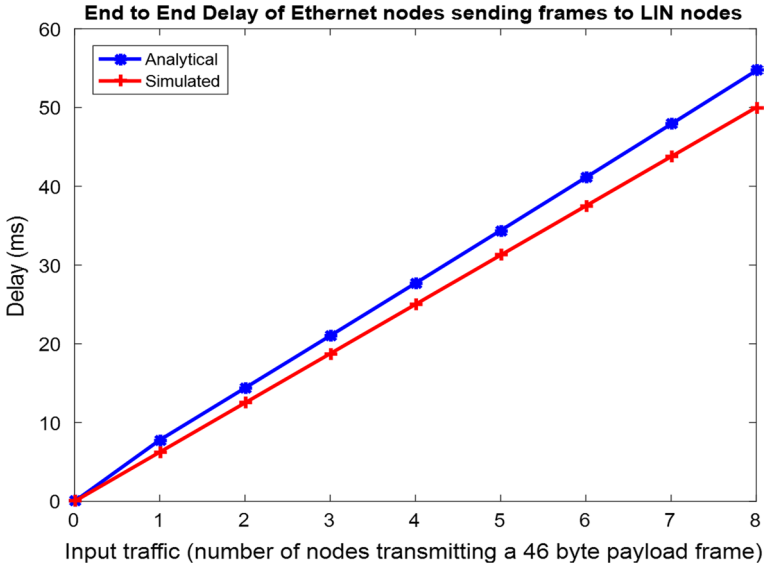


Fig. 46 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying the number of Ethernet nodes sending a 46-byte payload frame containing 8 bytes of LIN data to the LIN network segment. (Color figure online)

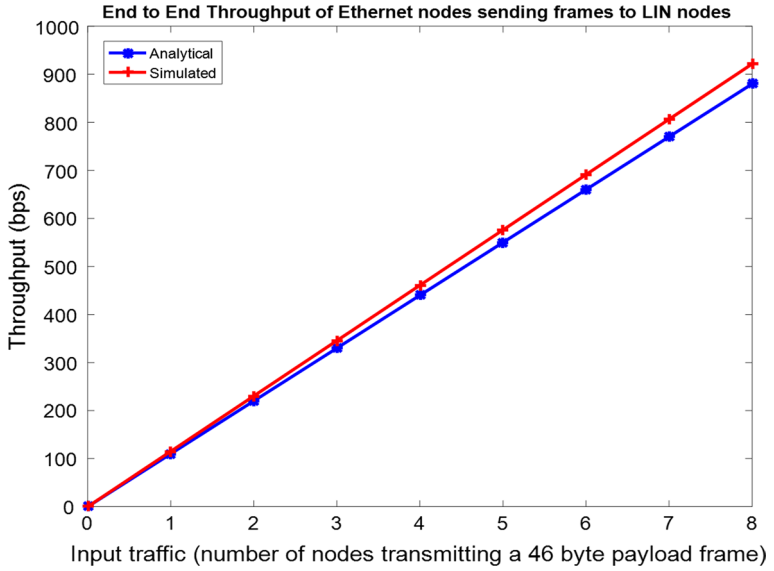


Fig. 47 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying the number of Ethernet nodes sending a 46-byte payload frame containing 8 bytes of LIN data to the LIN network segment. (Color figure online)

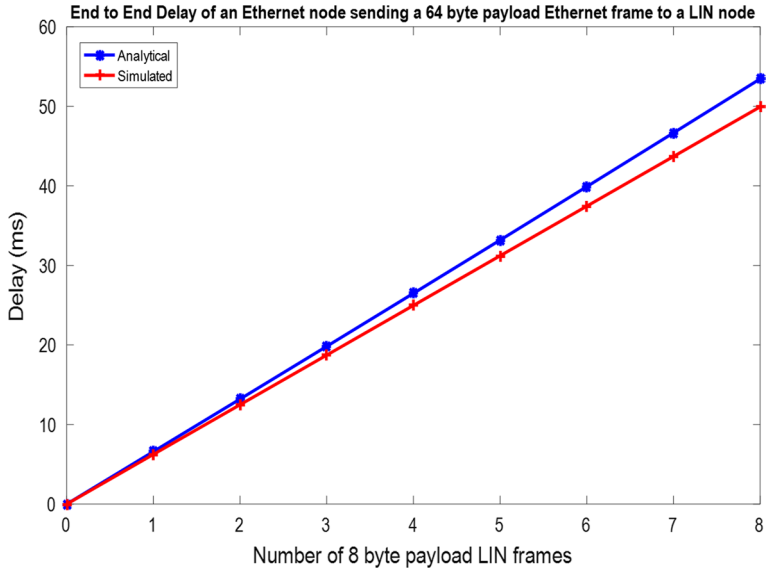


Fig. 48 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for delay when varying payload size (in steps of 8 bytes) of LIN data inside a 64-byte payload Ethernet frame sent from an Ethernet node to a LIN node. (Color figure online)

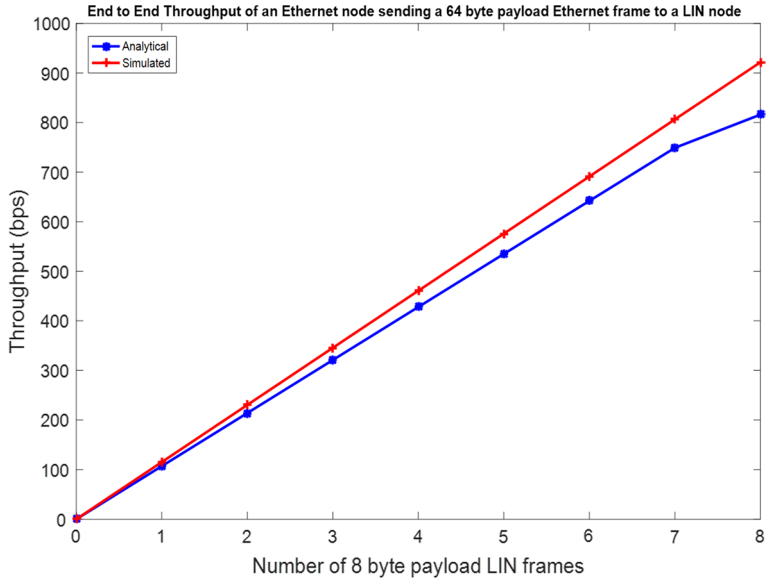


Fig. 49 One-way end-to-end analytical model results (blue graph) and the simulation results (red graph) for throughput when varying payload size (in steps of 8 bytes) of LIN data inside a 64-byte payload Ethernet frame sent from an Ethernet node to a LIN node. (Color figure online)

6.4.2 CAN-LIN End-to-End Performance Analysis

Figures 26 and 27 show the results of the analytical models and simulations for delay and throughput for route 1b when the payload is varied. The results for the variable node number case are shown in Figs. 28 and 29 for delay and throughput, respectively. This path is 1a reverse path. Since the maximum payload size of LIN and CAN is the same (8 bytes), no fragmentation is required.

6.4.3 CAN-ETHERNET End-to-End Performance Analysis

Figures 30 and 31 show the results of the analytical models and simulations for delay and throughput of route 2a for the variable payload case. For the variable node number case, the results are shown in Figs. 32 (delay) and 33 (throughput).

6.4.4 ETHERNET-CAN End-to-End Performance Analysis

For the path labeled 2b, Figs. 34 and 35 show the simulation and analytical model results for delay and throughput in the variable payload scenario. The results for the scenario where the node number is varied are shown in Figs. 36 and 37 for delay and throughput, respectively. Finally, and because fragmentation could occur in this path, Figs. 38 and 39 show the results when this happens. This occurs when the payload of

Table 11 Frame transmission configurations for heterogeneous network scenarios to evaluate one-way end-to-end delay and throughput performance

End-to-end delay and throughput model validation and performance analysis									
	Transmission case		Variation			Validation			
						Correlation		Root Mean Square Error	
Path Label	Single Frame	Multi Frames	Frame Payload	Number of Frames	Fragmentation	Delay (s)	Th (bps)	Delay (s)	Th (bps)
1a	✓		1–8 bytes	1		0.9918	0.9916	0.0007	5.17
1a		✓	8 bytes	1–8		0.9959	0.9916	0.0022	41.496
1b	✓		1–8 bytes	1		0.9917	0.9911	0.0003	11.82
1b		✓	8 bytes	1–8		0.9958	0.9997	0.0022	77.12
2a	✓		1–8 bytes	1		0.9991	0.9996	0.00001	23.45
2a		✓	8 bytes	1–8		0.9937	0.9999	0.00011	65.39
2b	✓		1–8 bytes	1		0.9990	0.9996	0.00003	23.45
2b		✓	8 bytes	1–8		0.9937	0.9999	0.0001	65.39
2b	✓		1–8 bytes	1	1–8	0.9874	0.9999	0.00019	65.39
3a	✓		1–8 bytes	1		0.9918	0.9912	0.0003	5.38
3a		✓	8 bytes	1–8		0.9999	0.9999	0.003	24.75
3b	✓		1–8 bytes	1		0.9917	0.991	0.0003	5.38
3b		✓	8 bytes	1–8		0.9999	0.9999	0.003	24.75
3b	✓		1–8 bytes	1	1–8	0.9999	0.9992	0.002	47.8

Table 12 Delay and Throughput one-way end-to-end performance results

Delay and Throughput end-to-end Performance Results						
Tx Node	Rx Node	Varying payload size	Varying number of nodes (transmission of multiple frames)	Frame fragmentation (1–8 frames of 8-bytes payload)	Maximum Delay (analytical/simulated) in ms	Maximum Throughput (analytical/simulated) in bps
LIN	CAN	✓			7.4/6.2	123.8/115.2
LIN	CAN		✓		47.5/49.9	990.7/921.6
LIN	Ethernet	✓			7.5/6.3	123.8/115.2
LIN	Ethernet		✓		54.7/60	880/921.6
CAN	LIN	✓			6.7/6.2	132/115.2
CAN	LIN		✓		47.2/49.9	1056.5/921.6
CAN	Ethernet	✓			0.5158/0.51	2484/2500
CAN	Ethernet		✓		2/2	856/966
Ethernet	CAN	✓			0.4572/0.51	2484/2500
Ethernet	CAN		✓		2/2	856/966
Ethernet	CAN			✓	/1.9	856/966
Ethernet	LIN	✓			6.7/6.3	124/115.2
Ethernet	LIN		✓		54.7/50	880/921.6
Ethernet	LIN			✓	53.5/49.9	816/921.6

the Ethernet frame (maximum size of about 1500 bytes) is greater than the data size that can be handled by the CAN frame (8 bytes).

6.4.5 LIN-ETHERNET End-to-End Performance Analysis

For path 3a, Figs. 40 and 41 show delay and throughput simulation and analytical results for the variable payload scenario. For the variable node number scenario, the results are shown in Figs. 42 and 43 for delay and throughput, respectively.

6.4.6 ETHERNET-LIN End-to-End Performance Analysis

Finally, for path 3b, fragmentation can occur. Figures 44 and 45 show the results for the variable payload scenario. For the variable nodes number scenario, the results are shown in Figs. 46 and 47, for delay and throughput, respectively. Finally, the results for the fragmentation scenario are shown in Fig. 48 for delay and Fig. 49 for throughput.

Table 11 gives a summary of the correlation and the RMSE values obtained for the different traffic flow cases presented in Figs. 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48 and 49.

Referring to Table 11, the correlation values obtained for the different cases are very similar, whereby the lowest correlation value for the delay was 0.9874, and the highest was 0.9999. The lowest correlation value for performance was 0.9910, and the highest was 0.9999. This implies that the graphs of the delay and throughput analytical models compared to the chart of the simulation results are almost identical. Meanwhile, there is a negligible difference between the analytical delay models and the delay simulation results for the obtained root mean square error values. For example, for path 1a concerning the transmission case: (i) Single frame, the RMSE was 0.68 ms, and (ii) for multiple frames, it was 2.2 ms. All other RMSE values obtained were corroborated as insignificant when comparing the analytical delay models and the delay simulation results. Therefore, the estimated simulation delay is a very close fit or approximation to the measured delay from the analytical model. On the other hand, the mean square error values obtained by comparing the analytical throughput model and the throughput simulation results are also negligible. For example, the RMSE single-frame case was 1014 bps (1.014 Kbps), and the multi-frame case was 7457 bps (7.457 Kbps). As a result, the estimated throughput simulation is almost a perfect fit and an acceptable approximation to the throughput analytical model for the different traffic flow cases. Comparing the end-to-end unidirectional maximum delay results from Table 12 with the delay requirements for different vehicle application use cases presented in [10], the maximum delay obtained for the different traffic flow cases is 54.7 ms (end-to-end delay from LIN nodes to Ethernet nodes and vice versa) which is half the maximum delay (100 ms) acceptable for the most demanding vehicular application use case.

The results presented in Table 12 are obtained considering a CAN bit rate of 500 Kbps, a Lin bit rate of 19,200 Kbps, an Ethernet bit rate of 100 Mbps, a LIN frame payload size of 8 bytes, a 46-byte payload Ethernet frame and a total of 8 frames sent from the LIN network to the Ethernet network and vice versa. From table 12, it is observed that the maximum throughput was for the traffic flow from CAN to Ethernet and vice versa for the variable payload size of the frame resulting in 2484/2500 bps (analytical/simulation). That throughput was conditioned to the performance experienced in the segment of the CAN network, which turned out to be the minimum of all the different throughput along the traffic flow path. In addition,

the results indicated that when the end-to-end traffic flow path involves a LIN segment, the maximum achievable throughput is conditioned to the throughput of the LIN network segment involved. Furthermore, for any end-to-end unidirectional traffic flow that does not include a LIN network segment, the results indicated that the throughput of the involved CAN network segment limits the maximum throughput.

7 Conclusion

Layer 2 modeling and analysis of an intra-vehicular network consisting of LIN, CAN, and Ethernet networks interconnected by bridges were performed and presented. Using the correlation as an evaluation metric to compare the analytical and simulation results of delay and throughput of the homogeneous network domains (CAN, LIN, and Ethernet), the mean correlation value obtained was 0.9983 for the delay (between the range 0.9911 and 0.9999), and 0.9932 for throughput (between the range of .9911 and 0.9999). For the heterogeneous end-to-end network scenarios, the mean correlation value obtained was 0.9951 for delay (between the range of 0.9874 and 0.9999) and 0.9973 for throughput (between the range of 0.9910 and 0.9999). The different validation cases show that the heterogeneous network configurations for LIN, CAN, and Ethernet are within the performance thresholds defined for the vehicular application use cases presented in [10]. However, if these thresholds are modified, the delay and throughput results can be verified using the proposed analytical models to validate in which cases they are met and can support the performance requirements for different network configurations.

Funding This work is supported in part by the National Council of Science and Technology (CONACyT) through the master (CVU 778010) and doctoral (CVU 854262) scholarship programs.

Data Availibility No data set was generated during the development of the presented work.

Declarations

Competing interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Wang, Y., Tian, D., Sheng, Z., Jian, W. (2017). Connected vehicle systems: Communication, data, and control. CRC Press, Florida. https://books.google.com.mx/books?id=5xIwDwAAQBAJ
2. Mathur, R., Saraswat, R., & Mathur, G. (2018). An analytical study of communication protocols used in automotive industry. *International Journal of Engineering Research & Technology (IJERT), 2*, 287–292.

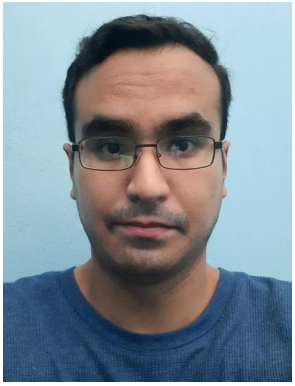
3. Navet, N., Simonot-Lion, F. (2017). In-vehicle communication networks: A historical perspective and review. In *Industrial communication technology handbook, second edition*. https://doi.org/10.1201/b17365.
4. Lee, E. A., Seshia, S. A. (2017). *Introduction to embedded systems: A cyber-physical systems approach*. MIT Press, Massachusetts. https://books.google.com.mx/books?id=chPiDQAAQBAJ.
5. Paul, A., Chilamkurti, N., Daniel, A., Rho, S. (2016). *Intelligent vehicular networks and communications: Fundamentals, architectures and solutions*, pp. 1–227.
6. Khoukhi, L., Xiong, H., Kumari, S., & Puech, N. (2021). The internet of vehicles and smart cities. *Annals of Telecommunications, 76*(9), 545–546. https://doi.org/10.1007/s12243-021-00891-7
7. Boukhalfa, F., Hadded, M., Muhlethaler, P., & Shagdar, O. (2021). Performance evaluation of an active signaling based time-slot scheduling scheme for connected vehicles. *Annals of Telecommunications, 76*(5), 363–374. https://doi.org/10.1007/s12243-020-00818-8
8. Dahi, S., & Jemili, I. (2021). A novel collision avoidance scheme for smart parking. *Annals of Telecommunications, 76*(9), 699–716. https://doi.org/10.1007/s12243-021-00878-4
9. Butt, T. A., Iqbal, R., Shah, S. C., & Umar, T. (2018). Social internet of vehicles: Architecture and enabling technologies. *Computers & Electrical Engineering, 69*, 68–84. https://doi.org/10.1016/j.compeleceng.2018.05.023
10. Zheng, K., Zheng, Q., Chatzimisios, P., Xiang, W., & Zhou, Y. (2015). Heterogeneous vehicular networking: A survey on architecture, challenges, and solutions. *IEEE Communications Surveys Tutorials, 17*(4), 2377–2396. https://doi.org/10.1109/COMST.2015.2440103
11. Tuohy, S., Glavin, M., Jones, E., Trivedi, M., Kilmartin, L. (2013). Next generation wired intra-vehicle networks, a review. In *2013 IEEE intelligent vehicles symposium (IV)*, pp. 777–782 . https://doi.org/10.1109/IVS.2013.6629561.
12. Umair, A. (2018). Communication technologies and network protocols of automotive systems. *Advances in Networks, 6*(1), 48 . https://doi.org/10.11648/j.net.20180601.15.
13. Jin, Y., Kwak, D., & Kwak, K. (2017). Performance evaluation of intra-vehicle wireless sensor network systems. *International Journal of Heavy Vehicle Systems, 24*, 158. https://doi.org/10.1504/IJHVS.2017.083086
14. Huang, J., Zhao, M., Zhou, Y., & Xing, C.-C. (2019). In-vehicle networking: Protocols, challenges, and solutions. *IEEE Network, 33*(1), 92–98. https://doi.org/10.1109/MNET.2018.1700448
15. vehicle Technical Committees, S. G. *Vehicle architecture for data communications standards*. https://www.sae.org/works/committeeResources.do?resourceID=386632.
16. Brown, A. (2011). *In-vehicle networking technology for 2010 and beyond (2010-01-0687)*, pp. 195–207.
17. Zeng, W., Khalid, M. A. S., & Chowdhury, S. (2016). In-vehicle networks outlook: Achievements and challenges. *IEEE Communications Surveys Tutorials, 18*(3), 1552–1571. https://doi.org/10.1109/COMST.2016.2521642
18. Karagiannis, G., Altintas, O., Ekici, E., Heijenk, G., Jarupan, B., Lin, K., & Weil, T. (2011). Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions. *IEEE Communications Surveys Tutorials, 13*(4), 584–616. https://doi.org/10.1109/SURV.2011.061411.00019
19. Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. *Mobile Networks and Applications, 19*(2), 171–209. https://doi.org/10.1007/s11036-013-0489-0
20. Bahga, A., Madisetti, V. (2014). Internet of Things: A hands-on approach. Arsheep Bahga & Vijay Madisetti, Virginia . https://books.google.co.in/books?id=JPKGBAAAQBAJ.
21. Smith, I. G. (2012). *The Internet of Things 2012: New Horizons*. CASAGRAS2, Halifax, UK . https://books.google.com.mx/books?id=1mszlAEACAAJ.
22. Benevolo, C., Dameri, R., & D'Auria, B. (2016). Smart Mobility in Smart City: Action taxonomy, ICT intensity and public benefits, *11*, 13–28. https://doi.org/10.1007/978-3-319-23784-8_2.
23. Informatik, V.: Introduction to CAN. Vector e-learning.
24. CiA (CAN in Automation: CAN data link layers: Three generations. https://www.can-cia.org/can-knowledge/can/can-data-link-layers/.
25. Natale, M. D., Zeng, H., Giusto, P., & Ghosal, A. (2012). *Understanding and using the controller area network communication protocol: Theory and practice* (1st ed.). New York: Springer.
26. Controller Area Network (CAN) Overview (2023). https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/controller-area-network-can-overview.html.
27. XNET NI-XNET Hardware and Software Manual.
28. Informatik, V.: Introduction to LIN. Vector e-learning

29. Hank, P., Müller, S., Vermesan, O., Van Den Keybus, J. (2013). Automotive ethernet: In-vehicle networking and smart mobility. In *2013 Design, automation test in Europe conference exhibition (DATE)*, pp. 1735–1739 . https://doi.org/10.7873/DATE.2013.349.
30. Kozierok, C. M., Correa, C., Boatright, R. B., Quesnelle, J. (2014). Automotive Ethernet: Definitive guide. Intrepid Control Systems, Michigan. https://books.google.com.mx/books?id=g1zToQEACAAJ.
31. Vector Informatik: Introduction to Automotive Ethernet. https://elearning.vector.com/vl_automotive_ethernet_introduction_en.html. Last accessed: 19 Dec 2020.
32. Barghi, H., Bredeson, J. G., Cronenwett, W. (1990). Throughput and delay analysis for a new efficient csma/cd based protocol. In *[1990] Proceedings. The Twenty-Second Southeastern Symposium on System Theory*, pp. 209–214 . https://doi.org/10.1109/SSST.1990.138140.
33. Inc., G.E.G.F.I.P. (2009). Switched ethernet latency analysis.
34. Gebali, F. (2008). *Analysis of computer and communication networks*. https://doi.org/10.1007/978-0-387-74437-7.
35. Hbaieb, A., Rhaiem, O. B., Chaari, L. (2018). In-car gateway architecture for intra and inter-vehicular networks. In *2018 14th International wireless communications and mobile computing conference (IWCMC)*, pp. 1489–1494 . IEEE
36. Seifert, R., Edwards, J. (2008). *The all-new switch book: The complete guide to LAN switching technology*. Wiley, London.
37. Mahmud, S. M., & Alles, S. (2005). In-vehicle network architecture for the next-generation vehicles. *SAE Transactions, 114*, 283–302.
38. Kim, J. H., Seo, S., Hai, N., Cheon, B. M., Lee, Y. S., & Jeon, J. W. (2015). Gateway framework for in-vehicle networks based on can, flexray, and ethernet. *IEEE Transactions on Vehicular Technology, 64*(10), 4472–4486. https://doi.org/10.1109/TVT.2014.237147
39. Rejeb, N., Karim, A., Salem, A. K., & Ben Saoud, S. (2016). Heterogeneous embedded network evaluation of can-switched ethernet architecture. *International Journal of Computer Science and Information Security, 14*, 280. https://doi.org/10.6084/m9.figshare.3153988
40. Thiele, D., Schlatow, J., Axer, P., & Ernst, R. (2016). Formal timing analysis of can-to-ethernet gateway strategies in automotive networks. *Real-Time Systems, 52*(1), 88–112. https://doi.org/10.1007/s11241-015-9243-y
41. López, A. (2014). Modeling and development of multibridges for heterogeneous networks modeling and development of multibridges for heterogeneous networks. Ph.D. thesis, CINVESTAV Guadalajara Unit.
42. Ekiz, H., Kutlu, A., Baba, M. D., Powner, E. T.: Design and implementation of a can/Ethernet Bridge, pp. 1–10.
43. Zeng, H., Di Natale, M., Giusto, P., & Sangiovanni-Vincentelli, A. (2009). Stochastic analysis of can-based real-time automotive systems. *IEEE Transactions on Industrial Informatics, 5*(4), 388–401. https://doi.org/10.1109/TII.2009.2032067
44. Li, F., Wang, L. F. (2014). Performance analysis of controller area network. In *Vehicle, mechatronics and information technologies II*. Applied Mechanics and Materials, vol. 543, pp. 3499–3502. Trans Tech Publications Ltd, Beijing . https://doi.org/10.4028/www.scientific.net/AMM.543-547.3499.
45. Di Natale, M., Zeng, H. (2010). System identification and extraction of timing properties from controller area network (can) message traces. In *2010 IEEE 15th conference on emerging technologies factory automation (ETFA 2010)*, pp. 1–8. https://doi.org/10.1109/ETFA.2010.5641332.
46. LIN Specification Package, Revision 2.0 (2003)
47. Rylander, A., Wallin, E. (2003). LIN—Local Interconnect Network—for use as sub-bus in Volvo trucks. Master's thesis, Chalmers University of Technology, Göteborg.
48. Ahlmark, M. (2000). Local interconnect network (lin)—packaging and scheduling.
49. Guzmán Mercado, J. I. (2016). Event driven lin protocol state machine using S12X microcontroller. Master's thesis, Instituto Tecnológico y de estudios superiores de occidente.
50. Gao, P., Li, D., Zhang, Y., Tan, L., Yang, L., Niu, B., Ning, Z., He, L., Yan, S. (2023). Verification and testing system for local interconnect network bus. In *2023 5th International conference on electronic engineering and informatics (EEI)*, pp. 334–339 . IEEE.
51. Lieder, R. (2017). Gateway processor evolution in automotive networks. The international CAN Conference (iCC), 6.
52. Zinner, H., Kleineberg, O., Boiger, C., Grzemba, A. (2012). Automotive requirements and definitions.
53. Tuohy, S., Glavin, M., Hughes, C., Jones, E., Trivedi, M., & Kilmartin, L. (2015). Intra-vehicle networks: A review. *IEEE Transactions on Intelligent Transportation Systems, 16*(2), 534–545. https://doi.org/10.1109/TITS.2014.2320605

54. hvanth, R., Valli, D., Ganesan, K. (2012). Design of an in-vehicle network (using LIN, CAN and FlexRay), gateway and its diagnostics using vector CANoe. *American Journal of Signal Processing, 1*(2), 40–45 . https://doi.org/10.5923/j.ajsp.20110102.07.

Guillermo Funes is a data analyst at Belize Telemedia Limited which is the national telecom provider in Belize. He currently leads the Capacity Management Planning team which focus on the monitoring, planning, optimization and expansion of mobile RAN (2G, 3G, 4G LTE), Mobile Core (CS, PS), transport (MPLS, DWDM) and fixed (FTTH) networks.During the 2016–2018 period he pursued his Master of Science in Electrical Engineering at CINVESTAV Unidad Guadalajara.His research project focused in the modelling, simulation and performance analysis of intra-vehicular heterogeneous networks. His areas of expertise are capacity planning of telecom networks, data analysis, modelling and forecasting. His areas of interest are Data Science, Data Analytics, Big Data, Telecom Networks, Machine Learning and Artificial Intelligence.

Mario Siller is a Principal Investigator at CINVESTAV Unidad Guadalajara. In that center is member of the Computer Science and Telecommunications Research Groups and Leads the Networked Systems Research Group. During 2015–2016 period he was a Visiting Associate Professor at MIT Media Lab and a Fulbright-García Robles Research Fellow. During this academic visit, he collaborated in the City Science area studying the emerging and aggregate complex cities behavior, from the Complex Systems Theory perspective. His research areas include Analysis and Modeling of Networked Systems, City Science, Distributed Systems, Computational Intelligence and IoT in multiple applications domains as Smart Cities, Health, Industry 4.0, Intelligent Transport Systems and Urban Agriculture.



Jorge Horta is a Ph.D. student in Telecommunications and member of the Networked Systems Research Group at the CINVESTAV Unidad Guadalajara. He obtained a degree in Computer Systems Engineering in Tecnológico Nacional de México, Instituto Tecnológico de Jiquilpan. He received his M.Sc. In Telecommunications at CINVESTAV Unidad Guadalajara, for the development of analytical models and simulations of networks that involved vehicular communication, cellular networks, and cloud networking. Currently his areas of study are Blockchain-based technologies and 5G networks as solutions to improve the security and performance of networks in the future.