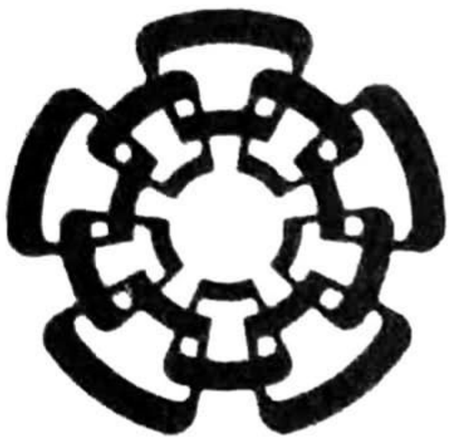
CT 809-551
DON. 7014



Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

Modelado y Desarrollo de Multipasarelas Para redes Heterogéneas

Tesis que presenta:
Abraham Jair López Villalvazo

para obtener el grado de:
Doctor en Ciencias

en la especialidad de:
Ingeniería Eléctrica

Director de Tesis:
Dr. Mario Angel Siller González Pico

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Abril de 2014

Modelado y Desarrollo de Multipasarelas para Redes Heterogeneas

Tesis de Doctorado en Ciencias
Ingeniería Eléctrica

Por:
Abraham Jair López Villalvazo
Maestro en Ciencias
CINVESTAV Unidad Guadalajara 2007-2010

Director de Tesis:
Dr. Mario Angel Siller González Pico

CINVESTAV del IPN Unidad Guadalajara, Abril, 2014.

Centro de Investigación y de Estudios Avanzados

del Instituto Politécnico Nacional

Unidad Guadalajara

Modeling and Development of Multibridges for Heterogeneous Networks

A thesis presented by:

Abraham Jair López Villalvazo

to obtain the degree of:

Doctor of Science

in the subject of:

Electrical Engineering

Thesis Advisors:

Dr. Mario Angel Siller González Pico

CINVESTAV del IPN Unidad Guadalajara,Guadalajara, Jalisco, April 2014

Modeling and Development of Multibridges for Heterogeneous Networks

Doctor of Science Thesis
In Electrical Engineering

By·
Abraham Jair López Villalvazo
Master in Science
CINVESTAV Unidad Guadalajara 2007-2010

Thesis Advisors:
Dr. Mario Angel Siller González Pico

CINVESTAV del IPN Unidad Guadalajara, April, 2014.

Resumen

Las redes heterogéneas permiten la creación de aplicaciones mucho más flexibles dado que la tecnología utilizada para la infraestructura de comunicaciones ya no es una limitante. Actualmente existen dispositivos que permiten interconectar dos segmentos de red que usan diferentes protocolos o incluso diferentes tecnologías. Para lograr este propósito, estos dispositivos simplemente encapsulan una trama de un dominio en el campo de datos del protocolo de destino.

La encapsulación no genera retardo extra por procesamiento ni por encolamiento en la pasarela misma, pero tampoco permite la mejor utilización del ancho de banda disponible. La utilización de la agregación de tramas permite que los protocolos de destino transporten más información del protocolo origen, aprovechando todo el ancho de banda disponible, pero genera retardos por encolamiento en el dispositivo de interconexión.

Dado todo esto, se propone el desarrollo de una pasarela para la interconexión de tres segmentos de red heterogéneos (802.11, 802.15.4 y CAN) que implemente la encapsulación y la agregación de tramas, pero también la fragmentación de tramas en los casos en que el protocolo de origen tiene un tamaño de trama mayor que la del protocolo destino.

El problema de obtener el mejor rendimiento en la utilización de ancho de banda y el menor retardo de fin a fin es analizado mediante un modelo matemático que representa cada una de las redes involucradas y los procesos propios de la pasarela. Para el análisis del desempeño de los segmentos de red, se propone la adecuación de dos modelos matemáticos existentes para los protocolos 802.11 y 802.15.4 y el desarrollo de un modelo matemático propio para el protocolo CAN. El modelado matemático de la pasarela se desarrolló utilizando teoría de teletráfico y teoría de colas.

Los modelos matemáticos se desarrollaron usando el enfoque de Cross-Layer Design, incluyendo características de capa 2 tales como el tamaño de trama, protocolo ARQ y MAC, y de capa 1 la tasa de error de bit y la codificación de canal.

Ésta investigación se enfoca en el modelado de la red heterogénea para estimar los mejores parámetros de comunicación para cada flujo de comunicaciones y específicamente en el modelado y desarrollo de una pasarela que interconecte segmentos de red heterogéneos implementando la encapsulación, agregación y fragmentación de tramas así como las configuraciones obtenidas de los modelos matemáticos antes mencionados que permitan obtener el mejor desempeño de fin a fin.

Abstract.

The heterogeneous networks allow creating more flexible applications given that the used communications infrastructure is not a restriction. Nowadays, existing devices allow the interconnection of two network segments using different communication protocols, even different transmission technology. To achieve this, these devices encapsulate a single frame from the origin protocol into the payload field of the destination protocol.

Single frame encapsulation does not generate processing delay, even queuing delay in the bridge, but do not allow the best use of the available bandwidth. Using frame aggregation allows destination protocol carrying more information from origin protocol, using the entire available bandwidth, but with greater delay generated by queuing processes in the interconnection device.

Given that, it is proposed the development of a bridge to interconnect three heterogeneous network segments (802.11, 802.15.4 AND CAN) that implements single frame encapsulation, frame aggregation, but also the frame fragmentation for the cases where the origin protocol has greater payload size than the destination protocol.

The delay-throughput tradeoff is analyzed through a mathematical model representing each network segment involved, for our case segments using 802.11, 802.15.4 and CAN and the bridging processes. For the performance analysis of network segments, it is proposed the adequacy of two existing mathematical models for 802.11 and 802.15.4 protocols. For the CAN protocol, it is developed a new mathematical model. The gateway mathematical model was developed using teletraffic and queuing theories.

Using a Cross-Layer design approach for its development, the mathematical model includes network characteristics from layer 2 such as frame size, ARQ and MAC protocols; and from layer 1, the bit error rate and channel codification.

This research is focused on the modeling of a heterogeneous network to achieve the estimation of the best communication parameters for each communications flow and specifically in the modeling and development of a heterogeneous bridge that interconnects different network segments implementing single frame encapsulation, frame aggregation and frame fragmentation, besides the implementation of the configuration parameters obtained from the mathematical models mentioned above, in order to obtain the best end to end network performance.

Index

Chapter 1. Introduction. 1
1.1 Motivation. 3
1.2 Problem definition. 3
1.3 Hypothesis 4
1.4 Research objectives 4
1.5 Thesis Organization and Contributions 5
Chapter 2. Theoretical Framework 6
2.1 Markov Chains 6
2.1.1 Limit of the power of the transition probability matrix P^n 7
2.1.2 Solving the balance equations. 7
2.2 Queuing and Switching Theory 7
2.2.1 The M/M/1/B queue 9
2.2.2 The $M/M^m/1/B$ 11
2.2.3 The $M^m/M/1/B$ 12
2.2.4 Switch fabric model 13
2.3 Bridging Theory 14
2.3.1 Bridging basics 15
2.3.2 Bridge architecture 15
2.3.3 Lookup Table design 17
2.4 Bridge classifications. 17
2.4.1 Forwarding Types 17
2.4.2 Buffer location 18
2.5 Bridging for different technologies 20
2.5.1 Translating Bridges 20
2.5.2 Encapsulating Bridges. 20
2.5.3 Encapsulating vs. Translating Bridges 20
2.5.4 Heterogeneous Bridging Issues 21
2.6 Network protocols. 22
2.6.1 IEEE 802.11 22
2.6.2 IEEE 802.15.4 23
2.6.3 Controller Area Network (CAN) 23
Chapter 3. State of the Art 25

3.1 MAC (Layer 2) based bridges 25
3.2 IP (Layer 3) based bridges 25
3.3 Middleware (Layer 7) based bridges 27
3.4 CLD based bridges 27
3.5 Specific bridging standards 27
3.6 CAN protocol modeling and performance analysis 28
3.7 802.11 protocol modeling and performance analysis 28
3.8 802.15.4 protocol modeling and performance analysis 29
Chapter 4. Proposed Bridge Design 30
4.1 Bridging Processes Formalization 30
Receiving process 30
Forwarding decision process. 31
Encapsulation Process 32
Frame Aggregation Process 32
Frame fragmentation process 32
4.2 Bridge Architecture Design 33
4.2.1 The receiving and address learning process 34
4.2.2 Lookup table management process 35
4.2.3 Protocol semantics module 35
4.2.4 Packet processing module 36
4.2.5 Statistics and adaptability module 37
4.3 Teletraffic Models 38
4.3.1 The 802.11 teletraffic model 39
4.3.2 The 802.15.4 teletraffic model 40
4.3.3 The CAN teletraffic model 43
4.3.4 Bridge teletraffic model 46
Chapter 5. Analysis and Results 48
5.1 CAN Teletraffic Model validation 48
5.2 CAN – 802.11 transmission experiments 51
5.2.1 CAN-802.11 transmissions analysis 52
5.2.2 802.11 – CAN transmissions analysis 53
5.3 Prototype of Heterogeneous Bridge implementation 54
5.4 Bridge performance analysis 55
5.4.1 Input buffer analysis 55

5.4.2 Output buffer analysis.....58
5.5 End to end transmission performance analysis.....64
5.5.1 End to end delay analysis.....64
5.5.2 End to end delay analysis.....66
Chapter 6. Conclusions and Future Work.....68
6.1 Conclusions......68
6.2 Future Work.....69
References.....70

Table Index

Table 1. Significance of diagonals of P 8
Table 2. Protocol addresses translation rules 36
Table 3. Meaning of the variables used in the 802.15.4 model 41
Table 4. Variables used in the CAN model and their meanings 45
Table 5. The message priorities and transmission periods per node (ECU) 48
Table 6. Transmissions configuration for end to end performance analysis 64

Figure Index

Fig. 1. a) Single Frame Encapsulation. b) Frame Aggregation. c) Frame Fragmentation. 2
Fig. 2. State transition diagram for M/M/1/B queue 9
Fig. 3. A 4 x 4 Crossbar switch fabrics a) CP matrix, b) N-way multiplexor for outputs 13
Fig. 4. The Bridge's main processes and its layer division 16
Fig. 5. The forwarding transmission times for both bridging modes. 18
Fig. 6. a) Translation of an 802.15.4 frame to 802.3 frame, b) Encapsulation of 802.15.4 into 802.3 payload 21
Fig. 7. MTU issues in the heterogeneous bridging process 22
Fig. 8. The CAN contention process. 24
Fig. 9. Frame formats for different protocols (802.15.4, 802.11, 802.3 and CAN). 24
Fig. 10. TCP/IP Overlay sensor networks 26
Fig. 11. Sensor networks overlay TCP/IP 26
Fig. 12. Interconnection of sensor and TCP/IP networks based on a bridge 27
Fig. 13. The general bridging process 33
Fig. 14. General architecture 34
Fig. 15. The receiving and address learning process 34
Fig. 16. Lookup Table management process 35
Fig. 17. Lookup Table format 35
Fig. 18. Protocol Semantics Module 37
Fig. 19. Packet processing module 37
Fig. 20. Statistics, Estimator and Adaptability Modules 38
Fig. 21. CAN Markov chain state diagram. 44
Fig. 22. The estimated, calculated and measured delay for message 0x0F1 varying the payload size 50
Fig. 23. Delay measurement of message 0x1F3 whit different number of nodes is transmitting simultaneously 50
Fig. 24. Comparison between measured and estimated delay for n simultaneous transmitting nodes. 50
Fig. 25. Comparison between measured and estimated loss probability varying the payload size. 51
Fig. 26. Comparison between measured and estimated throughput varying the payload size. 51
Fig. 27. Data flow in the prototype bridge implementation 55

Fig. 28. Throughput for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces 56
Fig. 29. Delay for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces 57
Fig. 30. Loss probability for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces 58
Fig. 31. Buffer size for the M/M/1/B input buffer varying the frame rate buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces 58
Fig. 32. Throughput for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) aggregating a) 802.15.4 and b) CAN traffic. 59
Fig. 33. Throughput for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic 59
Fig. 34. Delay for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) aggregating a) 802.15.4 and b) CAN traffic. 60
Fig. 35. Delay for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic 60
Fig. 36. Frame loss probability for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) forwarding 802.15.4 traffic. 61
Fig. 37. Frame Loss Probability for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic 61
Fig. 38. Queue size for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) forwarding 802.15.4 traffic. 61
Fig. 39. Queue Size for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic 62
Fig. 40. Throughput for the $M^m/M/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic 62
Fig. 41. Delay for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic. 63
Fig. 42. Frame Loss Probability for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic 63
Fig. 43. Queue Size for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic 63

Chapter 1. Introduction.

The massive deployment of information and communication technologies of the pervasive computing era, as observed by Mark Weiser [1], has raised a number of challenges such as: internetworking, energy consumption, mobility and portability. Also, has allowed the technology applications such as home automation and ambient intelligence are not just turning on-off lights appliances.

An Intelligent Environment (IE) can be defined as the integration of technology into an environment where the users interactively use it. According [2] the IE devices have three characteristics: they can be personalized, adaptive and anticipatory. In other words, an IE is an environment able to acquire information from itself and the users in order to improve the user experience in terms of comfort, security and/or productivity. Each implementation is constructed in a different way. However, in all systems the following elements are identified: (i) sensors which collect information for the system; (ii) the network which transports the information to the middleware; (iii) The middleware that processes the incoming information according to the context (iv); and (v) actuators which response to the environment and user changes. Applications such fire detection [3], house remote control [4], e-health, green houses are some examples.

This trend has reached even the automotive sector to improve mobility, comfort and safety. According to [5] the implementation of intelligent environments (IE) intended for car applications should consider information about the current situation inside and outside of the car, the physical and physiological driver conditions, driving intentions or needs and a friendly way to interact with users. In this sense a car can be an IE itself or as an extension of a particular IE. Furthermore, it can be part of a community of IEs or a standalone IE. The supported applications, services and tasks performed by a particular car delimit the class and case type.

The expanding field of applications for the IEs raises the need for interconnecting a broad variety of devices. The actuators and sensing applications available in the market uses different transmission technologies, a general condition in IEs where different network domains are involved. A network domain is defined as the network segment in which nodes use a common technology (protocol or standard using wired or wireless links) and can be dimensioned in terms of geography and distance. Heterogeneous networks (HN) are formed by two or more different domains. For example, an HN can be formed by a wired Ethernet local area network (LAN) connected to a Wi-Max (802.16) wireless metropolitan area network (MAN). Another HN can be produced from a Wi-Fi (802.11) wireless LAN connected to a Zigbee (802.15.4) personal area network (PAN) or control/sensor area network (CAN).

The interconnection of domains is made by a gateway or bridge which performs either (i) one to one framing translation or encapsulation, (ii) frame fragmentation or (iii) frame aggregation, (see Figure 1). In (i) no queue is required, however, a low throughput may be achieved whilst in (ii) and (iii) the

opposite is true. For our purpose, the frame encapsulation process corresponds to the action of allocating the source domain frame (including the header) into the payload field of the forwarding domain. Frame Fragmentation is the process of segmenting the source single protocol data unit (PDU) to fit in the forwarding protocol PDU, which has a smaller payload field. Frame aggregation refers to the action of using a PDU from the forwarding domain to transport two or more PDUs from the source domain.

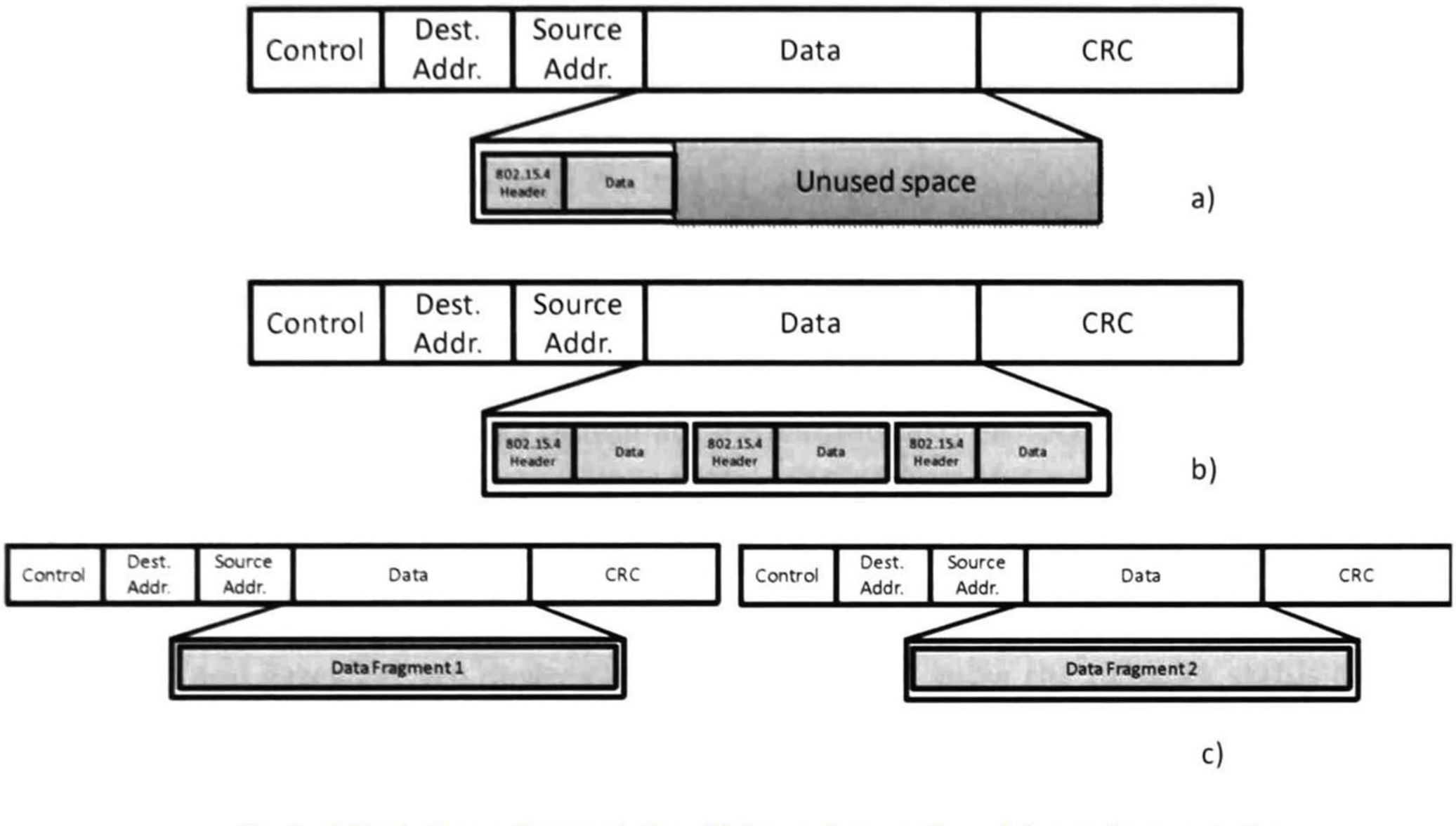


Fig. 1. a) Single Frame Encapsulation. b) Frame Aggregation. c) Frame Fragmentation.

In this research two performance metrics are considered: the end to end delay and throughput. The delay is the sum of all transmission delays, including the processing and queuing delays in the interconnecting bridges. The throughput represents the data transmission rate (only the payload, excluding headers) per time unit. To obtain the best throughput and delay in heterogeneous networks imply several variables, such as bit error rate, frame size and encapsulation, depending all on each domain's characteristics. Also, these performance metrics are affected by the bridging processes, when it performs frame encapsulation, aggregation or fragmentation. Given all involved domains have different conditions, is needed to analyze the communications in all directions (from each domain to all others and from all domains to each one).

Most of the interconnection solutions are based on a layered reference model such as the OSI or TCP/IP models. However, it has been proved that a cross layer design approach (CLD) [6] [7] [8] [9] can achieve better performance especially in pervasive computing environments. The CLD approach changes the layering paradigm established by OSI model; in [10] it is defined as "the exploit of information from multiple layers to jointly optimize performance of those layers". This approach has two key features: (i) all layers are able to communicate control information even if they are not adjacent and (ii) joining two or more layers is allowed [11].

1.1 Motivation.

The interconnection of devices using different technologies increases the possibilities to solve every day problems, even make the life more comfortable. Those applications require a reliable and efficient infrastructure allowing devices using different transmission technologies to interact in a transparent way for the final user. In other words, the intelligent environments provide an atmosphere that senses and responses to the presence of users and their needs.

Generally, when an intelligent environment project is planned, one of the design limitations is the supporting communications technology. The sensors, actuators and control nodes may use different protocol/technology to transmit the information. In this sense, for a smart house case, the interconnection requirements include the 802.11, and 802.15.4 protocols for the local area network and sensor and actuator nodes. In the other hand, the automotive industry requirements include the interconnection of CAN networks with other protocols such as 802.11 and 802.15.4.

Besides, the performance on HNs is affected by several factors such as framing, encapsulation, MAC (Medium Access Control) and ARQ (Automatic Repeat request) protocols, BER (bit error rate), frame aggregation/fragmentation and queuing in bridge. All of these factors affect the end to end delay and the throughput. Also, the users and/or network managers do not take into account the factors mentioned above, then the transmissions works, but in an inefficient way.

The bridges and gateways are devices with the capacity to know the network status and estimate the best configuration parameters. Being intermediate devices, they also have the capacity to notify these configurations to the network nodes in order to improve the network performance; however, to the best of our knowledge, this capacity has not been used.

1.2 Problem definition.

The interconnection devices based on protocol translation allows the communication between different network domains to set up a HN. The inherent characteristics of all network segments in this HN and the design requirements for the interconnection devices affect the forwarding process and the end to end network performance.

Three performance metrics are affected mainly within the bridge: The total delay is affected directly by two factors: i) required *store and forward* technique and ii) the size of queues used for receiving, processing and forwarding frames. Besides, the throughput is affected by the frame arriving frequency and size in addition of the output channel capacity.

Furthermore, the end to end performance is affected also by other factors regarding the heterogeneous network segments. The frame size affects directly the delay; the higher the frame size, the higher the delay. In the other hand, the throughput is affected by the bit error rate, the frame size and its header, and the used ARQ protocol.

This research addresses the heterogeneous network performance improvement by modeling and analyzing the network segments and the interconnection bridge based on a CLD approach. Also, it focuses on the design and development of a bridge that interconnects heterogeneous network

segments and contributes to the network performance improvement by identifying bridging processes and its behavior.

1.3 Hypothesis

It is our conjecture that the interconnection bridges impacts the end to end networks performance. Three specific metrics are affected by several factors concerning the bridging processes:

1. The Delay is affected by the queuing process and the buffer size used.
2. The Throughput is reduced when small buffers are used and
3. The frame loss (dropping) increases with this condition.

Besides, the network segment conditions, such as the BER, channel codification, MAC and ARQ protocols, frame size and framing process affect the end to end performance, decreasing the throughput and increasing the delay and loss probability

Therefore, the following hypotheses are stated:

> By setting the appropriate buffer size in each interface, the queuing delay in the bridge can be reduced and the throughput maintained in its maximum.
>
> The end to end throughput can be increased and the end to end delay reduced using frame aggregation when several frames have the same destination address.
>
> Using an optimum frame size in all network segments improves the end to end performance, specifically reducing the end to end delay and loss probability besides the increase of end to end throughput.

1.4 Research objectives

The main objective is to identify the factors affecting the end to end performance in heterogeneous networks by modeling mathematically the network segments and the interconnection device. Also, to design the necessary algorithms for the interconnection device and network configurations that improves the network performance.

The following are particular objectives for this research:

1. To design a translating bridge architecture allowing the transparent communication between 802.11, 802.15.4 and CAN networks.
2. To implement the bridging architecture in a test bed setup including the physical nodes for CAN, 802.11 and 802.15.4 networks.
3. To identify bridging processes affecting the network performance
4. To model the bridging processes using teletraffic theory.
5. To model the CAN network/transmissions using Markov chains and teletraffic theory with a CLD approach.
6. To adjust the 802.11 and 802.15.4 existing models to include the BER and ARQ protocol.
7. To estimate the network configuration settings that improve the performance, reducing the end to end delay and loss probability and increases the end to end throughput, following a CLD approach.

8. To implement the algorithms/configurations that improves the network performance following in the developed

All models will take into account: from layer 2: MAC and ARQ protocols, frame size, frame encapsulation, aggregation and fragmentation; from layer 1: Framing, bit error rate and modulation used.

1.5 Thesis Organization and Contributions

This document is divided into six chapters. Each chapter is briefly described below.

Chapter 2: In this chapter, the theoretical framework is described. This includes Markov chains and queuing theory used in the mathematical modeling and performance analysis; the bridging theory used in the bridge architecture design and finally a brief description of MAC and Physical layers of 802.11, 802.15.4 and CAN protocols.

Chapter 3: Is an analysis of previous works related to bridge design and analysis and protocol modeling and analysis.

Chapter 4: In this chapter, our work and contributions are described. First, the bridging processes formalization and the bridge architecture design are shown; also, the bridge and the protocols (802.11, 802.15.4 and CAN) teletraffic models are described.

Chapter 5: Our CAN Teletraffic model validation is presented in this chapter. Furthermore, performance analysis results for all protocols and the bridge are presented.

Chapter 6: This chapter shows the conclusions for the presented research and the future work.

The contributions of this research are:

A bridge architecture design to work in a heterogeneous network conformed by 802.11, 802.15.4 and CAN protocols.

The development of a teletraffic model for CAN protocol that allows analyzing and estimating throughput, frame loss and delay considering MAC and ARQ protocols from layer 2 and bit error rate from layer 1.

The adequacy of existing teletraffic models of 802.11 and 802.15.4 protocols to include the bit error rate and ARQ protocol.

The teletraffic models to analyze end to end delay and throughput for transmissions

- 802.11- 802.15.4
- 802.15.4 - CAN
- CAN - 802.11

Chapter 2. Theoretical Framework

2.1 Markov Chains

The Markov chains are a mathematical tool used to model dynamic systems that changes it state over the time. Modeling with Markov chains is simply, flexible and easy to compute.

A discrete time Markov chain (DTMC) is a sequence of random variables called *states* $\{X(n), n = 1,2,3, \dots\}$ with the Markov property. A discrete-time stochastic process $\{X(n), n = 1,2,3, \dots\}$ has the Markov property if

$$P(X(n+1) = j \mid X(n) = i, X(n-1) = i_{n-1}, \dots, X(0) = i_0) = P(X(n+1) = j \mid X(n) = i), \forall\, n$$

Which specifies that, given the current state *X(n)*, the probability distribution of the future state *X(n+1)* only depends on the previous *X(m)*, *m<n*.

Then, if *X(n)* is the state of a system at time *n*, the sequence $\{X(n), n = 1,2,3, \dots\}$ of states represent a dynamical system. The *state space S* is the set of all possible states. At any instance, the state of a Markov chain belongs to the state space *S*. The set *S* can be finite or countable infinite.

In the DTMC, the transition rule between states is specified by *transition probabilities*. The transition probability p_{ij} from state $i \in S$ to $j \in S$ can be written as follows:

$$p_{ij} = P\{X(n+1) = j | X(n) = i\}, i, j \in S$$

Where *X(n)* denotes the state of a Markov chain in time *n*. It is a conditional probability that the process is in state *j* at time *n+1* given that is in state *i* at time *n*. The transition probabilities of the entire system can be shown in a matrix form, which is called *transition probability matrix*. In general, this matrix can be represented by a square matrix *P*:

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{ij} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2j} & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ p_{i1} & p_{i2} & \cdots & p_{ij} & \cdots & p_{im} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mj} & \cdots & p_{mm} \end{bmatrix}$$

Where: $p_{ij} = P\{X(n+1) = j | X(n) = i\}, i, j \in S$ and m is the size of *S*. For a matrix to be a transition probability matrix, its elements must satisfy:

$$p_{ij} \geq 0, \sum_j p_{ij} = 1, \forall\, i \in S$$

The computing of the steady state distribution π of a Markov chain specified by (*S, P*) is called *solving the Markov Chain*. It can be shown that a DTMC with a finite state space has one and only one

steady state distribution if the process X_n can go from any state to any other state (not necessarily in one step). Markov chains with this property are said to be ***irreducible***.

The steady state vector π, can be understood as *a long term fraction of time being in a state*. From π, is easily computable any metric of our interest. There are two methods for calculating the steady state distribution of a DTMC: the limit of the power of the transition probability matrix P and the solution of the balance equations.

2.1.1 Limit of the power of the transition probability matrix P^n

This is the easiest way to calculate the steady state distribution through the power of the transition probability matrix *P*. The theoretical result is presented as follows: assume the existence of $\lim_{n\to\infty} P^n$ and let

$$\Pi = \lim_{n\to\infty} P^n$$

When each row vector of Π is identical to the first column is obtained a steady state distribution π of *P*. This limit exists for irreducible Markov chains such that the process X_n can go from one state to another in a finite number of steps.. Such Markov chains are said to be ***aperiodic***.

The n^{th} power P^n of *P* is called *n-step transition probability matrix* because its element p_{ij}^n corresponds to the probability that $X_{(m+n)} = j$ given that $X_m = i$ *or* $P[X_{(m+n)} = j | X_m = i]$

2.1.2 Solving the balance equations.

The second way of calculating the steady state distribution is by solving the following balance equations:

$$\pi = \pi P, \sum \pi_j = 1$$

Where π is a vector and P is a square matrix. Then, writing the i^{th} balance equation gives:

$$\pi_i = \sum_j \pi_j p_{ji} \quad or \quad \pi_i(1 - p_{ii}) = \sum_{i \neq j} \pi_j p_{ji}$$

The left term $\pi_i(1 - p_{ii})$ can be interpreted as the long term transition rate out of state *i* while the right term $\sum_{i \neq j} \pi_j p_{ji}$ is the long term transition rate into state *i*. The term *balance* in the balance equations comes from this observation that the rates in and out of state *i* are equal or balanced.

2.2 Queuing and Switching Theory

Queuing analysis is one of the most important tools for studying communication systems. It deals with *queues* where *customers* compete to be processed by shared *servers*. The *queue size* is the waiting room provided for the customers that have not been served yet plus the customers that are being served.

The objective of queuing analysis is to predict the system performance such as how many customers get processed per time step, the average delay of a customer endures before being served, and the

size of the queue or waiting room required. These performance measures have obvious applications in telecommunication systems and the design of hardware for such systems.

Queuing systems are special type of Markov chains in which customers arrive and lineup to be serviced by servers. Thus a queue is characterized by the number of arriving customers at a given time step, the number of servers, the size of the waiting area for customers and the number of customers that can leave in one time step.

Kendall's notation is used to describe a queuing system. This notation is represented as A/B/c/n/p where: A is the arrival statistics, B is the service or departure statistics, c is the number of servers, n is the queue size and p is the customer population size. The final two fields are optional and are assumed infinite if they are omitted. The letters A and B denoting arrival and server statistics are given the following notations: D for *deterministic*: the process has fixed arrival or service rates, M for *Markovian*, the process is Poisson or binomial, and G for *general* or constant time. A common practice is to attach a superscript to the letters A and B to denote multiple arrivals or batch service.

The state of a queuing system corresponds to the number of customers in the queue. As a type of Markov chains, the queuing systems also can be represented by a transition probability matrix. In specific, the diagonals of *P* reflect the queuing system characteristics. The Table 1 shows the significance of each diagonal of the matrix *P*.

Diagonal	Significance
Main	Probabilities queue will retain its size
1st upper	Probabilities queue size will decrease by one
2nd upper	Probabilities queue size will decrease by two
...	
1st lower	Probabilities queue size will increase by one
2nd lower	Probabilities queue size will increase by two
...	

Table 1. Significance of diagonals of P

There are several queuing systems which are related to the research presented in this thesis: M/M/1/B, M^m/M/1/B and M/M^m/1/B. The following metrics are used to measure the performance of the queuing systems:

Throughput (Th): It indicates the rate of frames leaving the bridge, expressed in frames per time step. This represents the average output traffic.

Efficiency (E): It represents the percentage of frames transmitted in one time step through the system relative to the total number of arriving customers or packets in one time step also. Essentially measures the effectiveness of the bridge at processing data present at the input.

Average number of frames (Q): represents the amount of frames stored in buffers and transiting on the bridge at a given time. This metric is measured in frames.

Waiting Time (W): represents the average number of time steps a frame spends in the bridge before leaving it and is measured in time steps.

2.2.1 The M/M/1/B queue

In this type of queue, at any step, at most one frame could arrive and at most one frame could leave. The queue has a finite size of *B*. at a certain time step, the probability of packet arrival is *a,* which is equivalent to a birth event or increase the queue length. The probability of the frame leaves the queue is *c*. This probability represents the ability to process the frames in the queue in one time step.

The number of frames in the system is the state of the queuing system. Thus, the queue is in state s_i when there are *i* frames in the queue. This queue system has a transition matrix which is shown as follows and state transition diagram shown in Figure 2.

$$P=\begin{bmatrix} f_0 & bc & 0 & \cdots & 0 & 0 & 0 \\ ad & f & bc & \cdots & 0 & 0 & 0 \\ 0 & ad & f & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & f & bc & 0 \\ 0 & 0 & 0 & \cdots & ad & f & bc \\ 0 & 0 & 0 & \cdots & 0 & ad & 1-bc \end{bmatrix}$$

Where $f_0 = 1\text{-}ad$ and $f=ac+bd$.

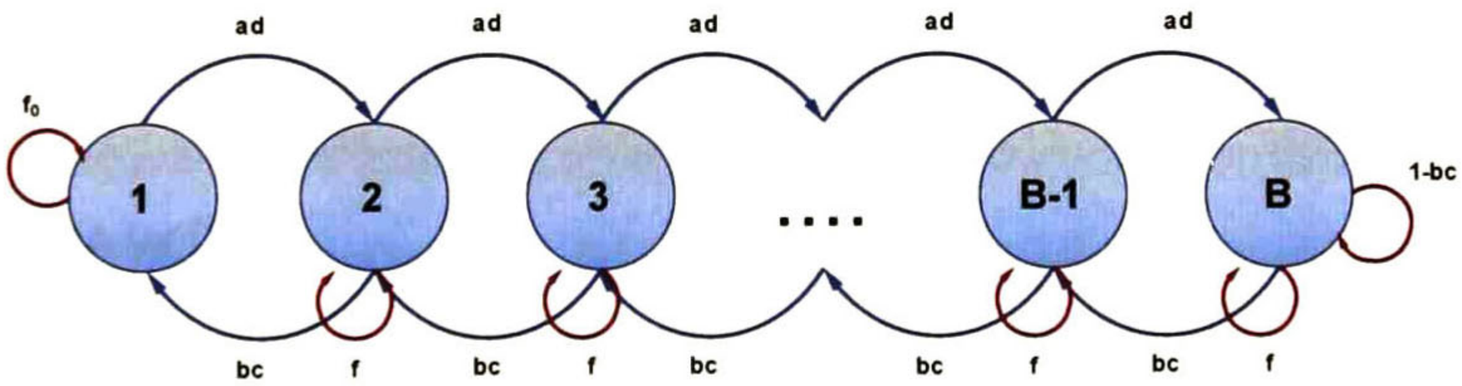


Fig. 2. State transition diagram for M/M/1/B queue

The difference equations for the state probability for the state probability vector are given by (47):

$$ad\ s_0 - bc\ s_1 = 0$$

$$ad\ s_0 - g\ s_1 + bc\ s_2 = 0$$

$$ad\ s_{i-1} - g\ s_i + bc\ s_{i+1} = 0 \qquad (47)$$

Where $g = ad + bc$ and s_i is the component of the distribution vector corresponding to state *i*. The solution to above equations is given as (48):

$$s_1 = \rho\ s_0$$

$$s_2 = \rho^2 s_0$$

$$s_3 = \rho^3 s_0 \qquad (48)$$

And in general: $s_i = \rho^i s_0$

Where $\rho = \left(\frac{ad}{bc}\right)$ is the distribution index for the M/M/1/B queue system and $0 \leq i \leq B$, then

$$s_0 \sum_{i=0}^{B} s_i = 1 \tag{49}$$

From which is obtained s_0, which is the probability that the queue is empty (50),

$$s_0 = \frac{1-\rho}{1-\rho^{B+1}} \tag{50}$$

And the equilibrium distribution for the other states is given by (51)

$$s_i = \frac{(1-\rho)\rho^i}{1-\rho^{B+1}} \quad 0 \leq i \leq B \tag{51}$$

Note that ρ for the finite-sized queue *can* be more than one. In that case the queue will not be stable (the queue will start to drop frames).

The throughput is given by (52):

$$Th = ac\, s_0 + \sum_{i=1}^{B} c\, s_i$$

$$Th = ac\, s_0 + c\,(1 - s_0)$$

$$Th = c(1 - B\, s_0) \tag{52}$$

The efficiency is given by (53):

$$E = \frac{Th}{a} \tag{53}$$

Frames are lost in this queue system when it is full and frames arrive but do not leave. The equation (54) is simply the probability that a packet is lost which equals the probability that the queue is full, a frame arrives and no frames can leave.

$$l = s_B\, a\, d \tag{54}$$

The frame loss probability L is the ratio of lost traffic relative to the input traffic (55)

$$L = \frac{l}{a} = s_B d \tag{55}$$

The average queue size expressed in frames is given by the equation (56)

$$Q = \sum_{i=0}^{B} i\, s_i \tag{56}$$

And the average time spent by a frame in queuing system is given by (57)

$$W = \frac{Q}{Th} \tag{57}$$

2.2.2 The M/M^m/1/B

In an M/M^m/1/B queue, the frames arrive at most one per time unit. The server can attend groups of average size of *m* per time unit and has a queue size limit of *B*. The batches of customer served represent the aggregation process in the bridge.

The frames arrive to the queue with probability of *a*, and *j* frames leave the queue when there are *i* frames in queue with probability of $c_{i,j} = \binom{i}{j} c^j d^{i-j}$. where *c* is the probability that a message departs and *d=1-c*.

The state transition matrix is an upper *(B+1) x (B+1)* Hessenberg matrix in which all the elements of subdiagonals 2, 3, ... are zero. The matrix has only *m* superdiagonals. For the case of *B*=6 and *m*=3 the matrix will have the next form:

$$P = \begin{bmatrix} q_0 & r_1 & s_2 & t_3 & 0 & 0 & 0 \\ p_0 & q_1 & r_2 & s_3 & t_4 & 0 & 0 \\ 0 & p_1 & q_2 & r_3 & s_4 & t_5 & 0 \\ 0 & 0 & p_2 & q_3 & r_4 & s_5 & y \\ 0 & 0 & 0 & p_3 & q_4 & r_5 & x_2 \\ 0 & 0 & 0 & 0 & p_4 & q_5 & x_1 \\ 0 & 0 & 0 & 0 & 0 & p_5 & x_0 \end{bmatrix}$$

These matrix elements are given by the general expressions (58):

$$\begin{aligned} p_i &= a\, c_{i+1,0} \\ q_i &= a\, c_{i+1,1} + b c_{i,0} \\ r_i &= a\, c_{i+1,2} + b\, c_{i,1} \\ s_i &= a \sum_{j=m}^{i+1} c_{i+1,j} + b c_{i,2} \\ t_i &= b \sum_{j=m}^{i} c_{i,j} \\ x_i &= c_{B,i} \\ y &= \textstyle\sum_{j=m}^{B} c_{B,j} \end{aligned} \tag{58}$$

Where *b=1-a*. The throughput, packet loss probability, average queue size and delay can be formulated as follows (59-62):

$$T = {a(1 - s_B)}/{T} \tag{59}$$

$$L = s_B c_{B,0} \tag{60}$$

$$Q = \sum_{i=0}^{B} i s_i \tag{61}$$

$$W = {}^{Q_{ec}}/_{T_{ec}} \tag{62}$$

Where s_B is the probability of full buffer, $c_{B,0}$ is the probability of full buffer and no messages leave the queue, and s_i are the probability of i messages in queue.

2.2.3 The $M^m/M/1/B$

In an $M^m/M/1/B$ queue, the frames can arrive in batches or clusters of average size of m and at most one of them could leave. The queue has a size limit of B and will be frames that cannot be served when the queue size is equal to B [15]. The group of customers represents the large frames that will be fragmented. The batch size is the incoming frame size divided by the outgoing frame size.

The transition matrix is a lower (B+1) x (B+1) Hessenberg matrix in which all the elements $p_{i,j}=0$ for $j>i+1$ and m subdiagonals as shown

$$P = \begin{bmatrix} x & y_0 & 0 & \cdots & 0 & 0 \\ y_2 & y_1 & y_0 & \cdots & 0 & 0 \\ y_3 & y_2 & y_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \cdot & \vdots & \vdots \\ y_B & y_{B-1} & y_{B-2} & \cdots & y_1 & y_0 \\ z_B & z_{B-1} & z_{b-2} & \cdots & z_1 & z_0 \end{bmatrix}$$

Where:

$$x = a_1\, c + a_0$$
$$y_i = a_i c + a_{i-1} d$$
$$z_i = a_i d + \sum_{k=i+1}^{m} a_k$$

And assuming $a_k = 0\ then\ k < 0;\ a_k = k > m$. Batches of m packets arrive to the queue with probability of a_m. The frames leave one at time with probability of c. The throughput (63), the loss probability (64), average queue length (65) and delay (66) are formulates as follows.

$$Th = c(1 - a_0 s_0) \tag{63}$$

$$L = 1 - \frac{c(1 - a_0 s_0)}{m\, a} \tag{64}$$

$$Q = \sum_{i=0}^{B} i\, s_i \tag{65}$$

$$W = {}^{Q}/_{T} \tag{66}$$

Where a_0 is the probability of no arrivals, s_0 is the probability of staying in the empty state, s_i the probability of i messages in the queue and T is the time interval.

2.2.4 Switch fabric model

The crossbar can be considered as a set of shared media (i.e. the columns of the crossbar network are associated with an output queue or link). All *N* columns work in parallel and accept traffic for all *N* inputs (rows in the crossbar). The Figure 3 shows two implementations of a 4 x 4 crossbar. The first implementation a) consists in an array of cross points (CP) connected in a grid way, the second b) uses an N-way multiplexer for each output to select the data from the input ports.

Suppose that two or more inputs request access to the same output, in that case, contention arises and some arbitration mechanism has to be implemented. An arbiter for each column must be implemented in order to control the output buffer or link access. This arbitration mechanisms are necessary when the crossbar are *input driven*, where the inputs issues the request to configure the cross points. In the *output driven* crossbar switches, the outputs initiate the issuing of requests for data from the inputs and this eliminates the end for the arbiters. This form of operation leads to much faster bridging operation [15].

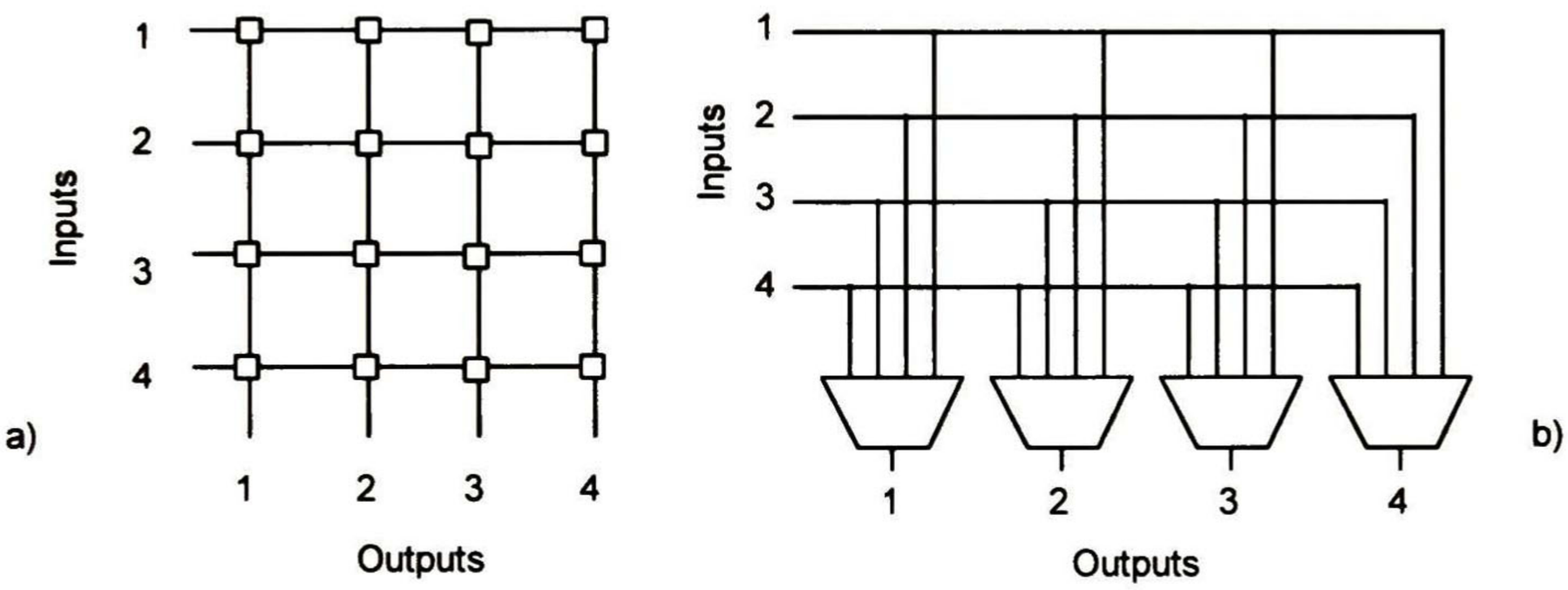


Fig. 3. A 4 x 4 Crossbar switch fabrics a) CP matrix, b) N-way multiplexor for outputs

Our implementation follows the output driven approach, then an information requester for each interface is implemented (see Figure 3.b).

Let assume a as the packet arrival probability at an input port of an N x N crossbar switch. The probability that a packet arrives at any output port is given by (67)

$$a' = {}^{a}/_{N} \tag{67}$$

This is because an arriving packet has an equal probability of requesting one of N output ports. Each output port deals with *N* users but the probability of packet arrival is reduced to ${}^{a}/_{N}$. The probability that *i* requests arrive at a time slot addresses to the tagged output port is given by (68):

$$p(i) = \binom{N}{i}\left(\frac{a}{N}\right)^{i}\left(1-\frac{a}{N}\right)^{N-1} \tag{68}$$

The input traffic destined for the tagged output port per time slot is given by (69)

$$N_{a(in)} = \sum_{i=0}^{N} ip(i) = a \tag{69}$$

The throughput of the tagged output port is equal to the output traffic of the tagged output port and is given by (70):

$$Th = \sum_{i=1}^{N} p(i) = 1 - P(0) \tag{70}$$

If *a'* is the traffic sent to the tagged output port, then substituting in the formula (70) is obtained:

$$Th = N_{a(out)}$$

$$Th = 1 - \left(1 - \frac{a}{N}\right)^{N} \tag{71}$$

The second term in the RHS is the probability that no packets are destined to the tagged output port.

The packet acceptance probability is given by (72):

$$p_a = \frac{N_{a(in)}}{N_{a(out)}} = \frac{1}{a}\left[1 - \left(1 - \frac{a}{N}\right)^{N}\right] \tag{72}$$

The delay of the crossbar network is defined as the average number of attempts to access the desired output. The probability that the packet succeeds after *k* tries is given by the geometric distribution (73):

$$p(k) = p_a(1 - p_a)^k \tag{73}$$

The average delay time is formulated in (74)

$$n_a = \sum_{k=0}^{\infty} k\, p(k)$$

$$n_a = \sum_{k=0}^{\infty} k\, p_a(1 - p_a)^k$$

$$n_a = \frac{1 - p_a}{p_a} \tag{74}$$

2.3 Bridging Theory

Bridges are *intermediate stations* that serve as relay devices. These devices that accepts frames at its inputs and forwards them to its outputs according to the information provided in the frame header and the bridge lookup table.

2.3.1 Bridging basics

A bridge has to perform several functions besides simply routing packets from its inputs to its outputs.

Forwarding: The device must be able to read the header of each incoming packet to determine which output link must be used to move the packet to its destination. It is obvious that an arriving packet cannot be routed until packet classification based on it header information has been performed. These forwarding decisions are done using lookup tables.

Traffic management: when too many frames are present in the network, *congestion* can take place. When congestion occurs in a bridge, the buffers become filled and frames start getting lost. This situation typically gets worse since the receiver will start sending negative acknowledgments and the sender will start retransmitting the lost frames. This leads to more frames in the network and the overall performance starts to deteriorate. *Traffic management* protocols must be implemented at the bridges to ensure that users do not exhaust the network resources.

Scheduling: bridges must employ scheduling for two reasons: to provide different quality of service to different type of users and to protect well-behaved users from misbehaving users that might take all the system resources. A scheduling algorithm might operate on a per-flow basis or it could aggregate several users into broad service classes to reduce the workload.

Congestion control: the bridge drops frames to reduce network congestion and to improve its efficiency. The bridge must select which packet to drop when the system resources become overloaded. There are several options such as dropping packets that arrive after the buffer reaches a certain level of occupancy (A.K.A. tail dropping). Another option is to drop packets from any place in the buffer depending on their tag or priority.

2.3.2 Bridge architecture

The bridge has two main architectural components [12]: **input/output interfaces** and **forwarding logic**. Depending on the type of bridge, the interfaces can provide interconnection to network segments using the same or different protocol/technology. The interfaces deals with individual frames and performs the following functions: accepts incoming frames on any of its N input ports and stores them in temporary buffers for processing their header information; stores the forwarding packets in the output queues; schedules the stored frames for transmission and performs the proper transmission. The input/output Interfaces perform all layer 1 and 2 transmission actions, such as encapsulation according the MAC and ARQ protocols, besides the proper encoding for transmission. This process contains software (for layer 2) and hardware components (for layer 1) to complete the frame reception and transmission.

The **forwarding logic** establishes the required paths between pairs of input and output ports. To determine the proper destination, the forwarding logic maintains a Lookup Table (LT). This table contains the source addresses and the input port for each frame that arrives to the bridge. The forwarding logic is divided in two elements which are responsible for (i) **switch fabric** and (ii) the maintenance of the LT. To decide if a frame must be forwarded or filtered, the switch fabric

compares the source address with the entries in the LT. If there is an entry in the LT matching with the source address and the input port, then the received frame is dropped because it means that the frame is part of the local traffic and does not need to be forwarded. Otherwise, the destination address is searched; if there is a matching entry with both the address and the input port, the frame is forwarded to the indicated port; else, the frame is flooded to all ports.

The maintenance of information (ii) requires several sub-processes (see Fig 4). The LT is a dynamic entity and is the key to proper bridge operation. The ***Learning Logic*** enables the bridge to acquire the information for the LT. The incoming frame is analyzed and MAC addresses and source port is gathered and stored in LT from each incoming communication and from the flooding process. The ***age verification process*** prevents the excessive growth of LT placing a time stamp on each entry. The *aging time* parameter controls the pruning of old entries from LT. The IEEE 802.1D standard [13] allows a range of aging times from 10 seconds to 1 million seconds (11.6 days) with recommended default of 300 seconds (5 minutes). Maintaining as short as possible the address table helps to make faster searches and to reduce the buffer needs. This process is a non-critical, low priority task and it can be done in the background. The bridges interconnecting networks with different technology requires a ***protocol translation process***, taking the information from source protocol/technology and delivering the frame in proper format for the forwarding protocol/technology. The ***filter and forwarding logic*** analyzes the destination address in the incoming frames, and using the LT, takes the forwarding decision. If the frame is destined to local traffic, it is discarded (or filtered) otherwise is forwarded to corresponding interface.

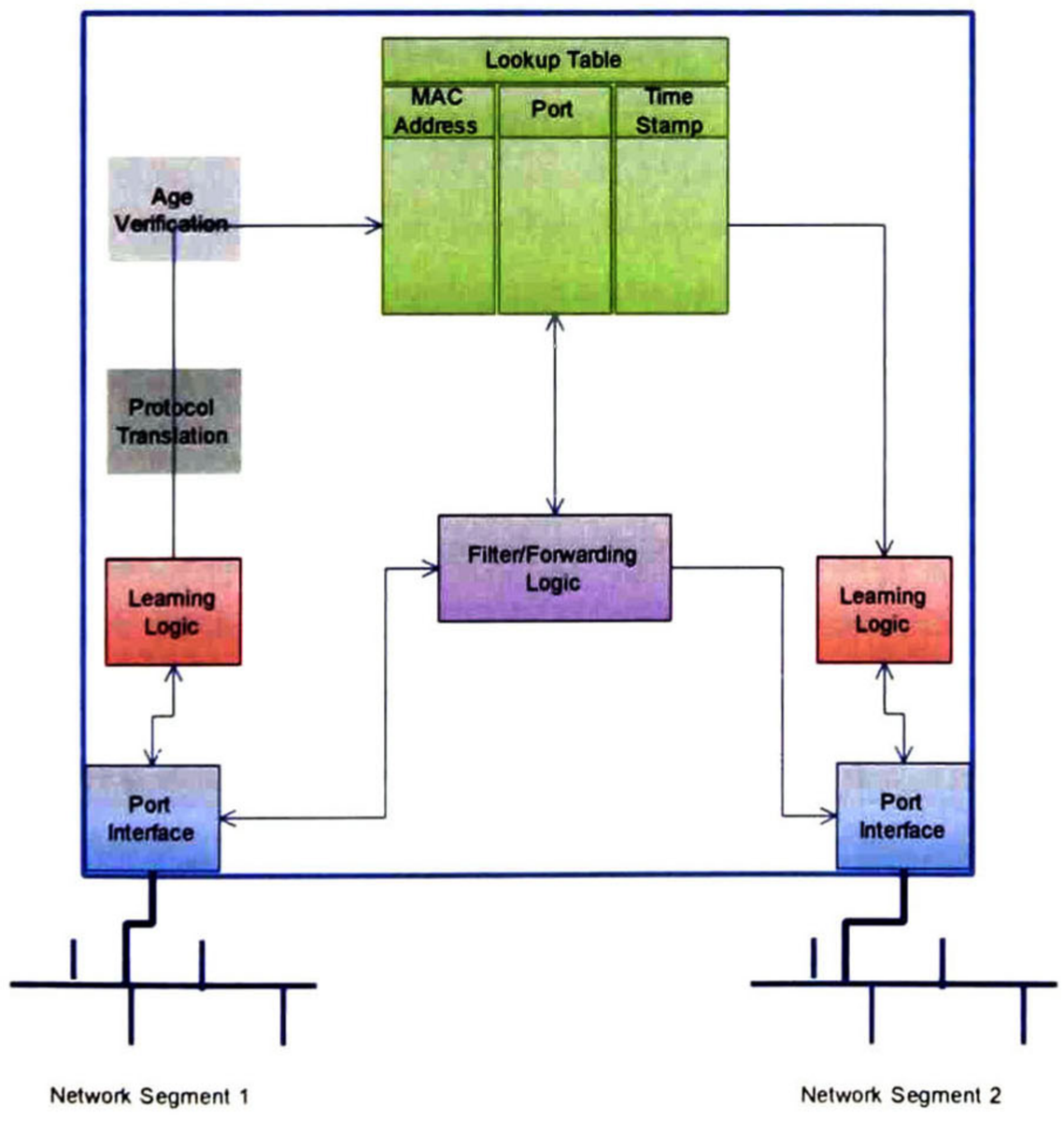


Fig. 4. The Bridge's main processes and its layer division

2.3.3 Lookup Table design

The lookup table stores information about which output ports should forward a packet that arrives at an input port. The data in the LT could be organized using any of the following approaches:

1. ***Heap storage***: store the data in a heap with no particular order using a random access memory. This approach is simple but searching for a particular data item requires searching the entire memory. Thus accessing the database is done one item at time using a slow search strategy which is not practical in high-speed switches.
2. ***Content-addressable memory*** (CAM): this is sometimes called associative memory or associative storage. This type of memory requires the user to provide a data word. The CAM will search in all memory content to see if the data word is stored. The content search is commonly implemented using a binary tree. Searching for a data item is done in a distributed fashion within the memory. This speeds up the search operation. CAM could be a small memory but complex in design and slow performance.
3. ***Hash tables***: here the input header information is compressed to a smaller number of bits (e.g., from 48 bits to 16 bits) using a hashing function to reduce the RAM size. The hashing function should be simple to implement without requiring much processing time.
4. ***Balanced tree***: the B-tree is a data structure that is used to efficiently build large dictionaries and implement the algorithms used to search, insert and delete keys from it. The advantage of using B-trees for constructing lookup tables is the small time required to search for the routing information.

2.4 Bridge classifications.

Two main characteristics of a bridge are the forwarding technique and the buffer location. The bridges can be classified by those functional characteristics.

2.4.1 Forwarding Types

The bridge type decides how to handle a frame when it arrives on a bridge interface. This decision directly affects the latency. There are two main bridging modes [14] (see Figure 5):

Cut-through: (or fast forward): When this type is used, the bridge only waits to read the destination address to take the forwarding decision and proceeds to forward the frame. This type reduces the latency because it begins to forward data as soon as it reads the destination address (does not require the use of buffers). The cons are that it cannot perform error checking without storing the frame in buffers.

Store-and-forward: Using this type, the frame must be received entirely to be forwarded. The bridge can perform a CRC and if the frame is correct, it is then forwarded. The cons are that the latency is increased because the buffering and CRC checking.

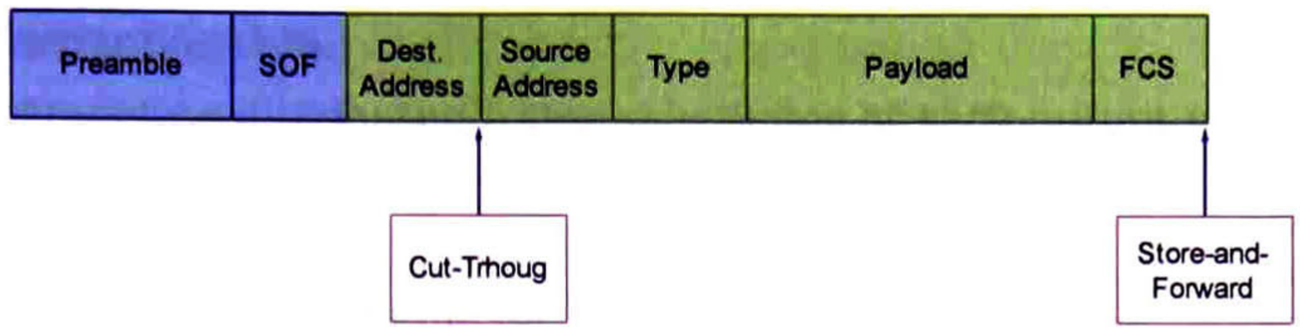


Fig. 5. The forwarding transmission times for both bridging modes.

2.4.2 Buffer location

An important characteristic of a bridge is the number and location of the buffers used to store the incoming traffic to be processed. The placement of the buffers and queues in a bridge is of very important since it will impact the bridge performance measures such as packet loss, speed, ability to support differentiated services, etc. [15].

Input Queue Bridge: each input port has dedicated FIFO (First In - First Out) buffer to store incoming packets. The arriving packets are stored at the tail of the queue and only move up when the frame at the head of the queue is forwarded through the forwarding logic to the correct output port. A controller at each input port classifies each packet by examining its header to determine the appropriate path through the forwarding logic.

The main advantages of input queuing are:

1. Low memory speed requirement
2. Distributed traffic management at each input port
3. Distributed table lookup at each input port
4. Support of broadcast and multicast does not require duplicating the data.

The main disadvantages are:

1. Head of line (HOL) problem
2. Difficulty in implementing data broadcast or multicast since this will further slowdown the bridge due to the multiplication of HOL problem.
3. Difficulty in implementing QoS
4. Difficulty in implementing scheduling strategies since this involves extensive communications between the input ports.

The HOL problem arises when the packet at the head of the queue is blocked from accessing the desired output port. This blockage could arise because the switch fabric cannot provide a path or if another packet is accessing the port.

Output Queue Bridge: To overcome the HOL limitations of input queuing, the standard approach is to change input queuing to output queues. An incoming frame is stored at a small input buffer and the controller must read the header information to determine which output queue is to be updated.

The main advantages of output queuing are:

1. Distributed traffic management.
2. Distributed lookup table at each input port.

3. Ease of implementing QoS
4. Ease of implementing distributed frame scheduling at each output port.

The main disadvantages are:

1. High memory speed requirements for the output queues
2. Difficulty of implementing data broadcast or multicast since this will further slowdown the switch due to the multiplication of HOL problem.
3. Support of broadcast and multicast requires duplicating the same data at different buffers associated with each output port.
4. HOL problem is still present since the switch has input queues.

Shared Buffer Bridge. This type of bridge employs a single common buffer in which all arriving packets are stored. This buffer enqueues the data in separate queues wich are located within one common memory. Each queue is associated with an output port. A flexible mechanism employed to construct queues using a regular RAM is to use the linked lists. Each linked lists is dedicated to an output port. The lengths of liked lists need not be equal and depend only on how many packets are stored in each liked list.

When a new frame arrives at an input port, the buffer controller decides which queue it should go to and stores the packet at any available location in the memory, then appends that frame to the linked list by updating the necessary pointers. When a packet leaves a queue, the pointer of the next packet now points to the output port and the length of the linked list is reduced by one.

The main advantages of shared buffering are:

1. Ability to assign different buffer space for each output port since the liked list size is flexible and limited only by the amount of free space in the shared buffer.
2. Distributed table lookup at each input port.
3. There is no HOL problem in shared buffer switch since each linked list is dedicated to one output port
4. Ease of implementing data broadcasts or multicast.
5. Ease of implementing QoS.
6. Ease of implementing scheduling algorithms at each linked list.

The main disadvantages are:

1. High memory speed requirements for the shared buffer.
2. Centralized scheduler function implementation which might slow down the bridge.
3. Support of broadcast and multicast requires duplicating the same data at different linked lists associated with each output port.
4. The use of a single shared buffer makes the task of accessing the memory very difficult for implementing scheduling algorithms, traffic management and QoS support.

Multiple input/output queues bridge: an arriving frame must be classified by the input controller at each input port to be placed in its proper input queue. A packet directed to a certain output port travel sthrough the forwarding logic, and the controller at each output port classifies them, according to their class of service, and places them in their output queue.

The advantages of multiple queues at the input and the output are the removal of HOL problem, distributed table lookup, distributed traffic management, and ease of implementation of QoS. The disadvantage of the multiple input and output queue bridge is the need to design a forwarding logic that is able to support maximum of $N^2 \times N^2$ connections simultaneously.

2.5 Bridging for different technologies

The main function of the bridge is to interconnect two or more network segments. When those segments use different technology, the bridge must be capable to manage different access control methods, frame formats (see Figure 6), frame semantics and data field length [16]. This heterogeneity increases the bridge complexity. To maintain the premise of relaying the frames between network segments transparently, the bridge must take into account that each network segment is independent and has their own access control and that the bridge does not have privileged status on the connected segments. To achieve this objective, two techniques can be used: Frame format translation and frame encapsulation.

2.5.1 Translating Bridges

Generally, a bridge transmits the frames using the format appropriate for each network segments. However, when a frame is received on one segment and must be forwarded to a different one, the bridge must perform a frame format translation. The translation consists in two parts [14]:

Mac Specific Formats: The bridge must map between the MAC specific fields, for instance the source and destination addresses. In some cases, a protocol structure have fields that in other do not exists, even, if the field exists in all protocols, it may be different in format or length, the semantics of the bits or the fields are different. Assuming this, some information or field must be reformatted, recalculated or generated for the new frame, and others may be eliminated. Here is where the protocol translation takes its place (see Figure 3a).

User Data Encapsulation: The bridge must map between the different methods used to encapsulate user data among the various network segments. The data encapsulation takes into account technology specific details of the MAC frame formats. The different ways in which user data can be encapsulated into the payload field (see Figure 3b).

2.5.2 Encapsulating Bridges.

There is an alternative to the Frame Format Translation technique when bridging dissimilar network segments. Instead of translate the received frame fields and/or format to forward it, an encapsulating bridge simply encapsulates the entire received frame into the payload field of the forwarding frame. Note that the header and all control information are included in the payload of the forwarding protocol and this increases the total overhead reducing the total throughput.

2.5.3 Encapsulating vs. Translating Bridges

Both techniques present pros and cons, and its use depends on the application and the network conditions. The following is a comparison between these techniques showing its pros and cons.

The encapsulation avoids the need to transform the frame and the original structure is preserved, reducing the delay and processing requirements (see Fig. 6b). The translation requires more resources, but the total overhead is not increased, increasing the throughput and at the cost of slightly increasing the delay due to the frame storage and processing (see Fig. 6a).

Since encapsulation bridges forward frames only to specific nodes, the bridge only needs to receive the frames that actually require forwarding. The translating bridge needs to analyze all frames in the local segment and decide which ones need to be forwarded, taking time and processing resources. The advantage of translating bridges is that they can interconnect any network type, meanwhile the encapsulating bridges only can forward to other network with the same technology as the forwarding interface.

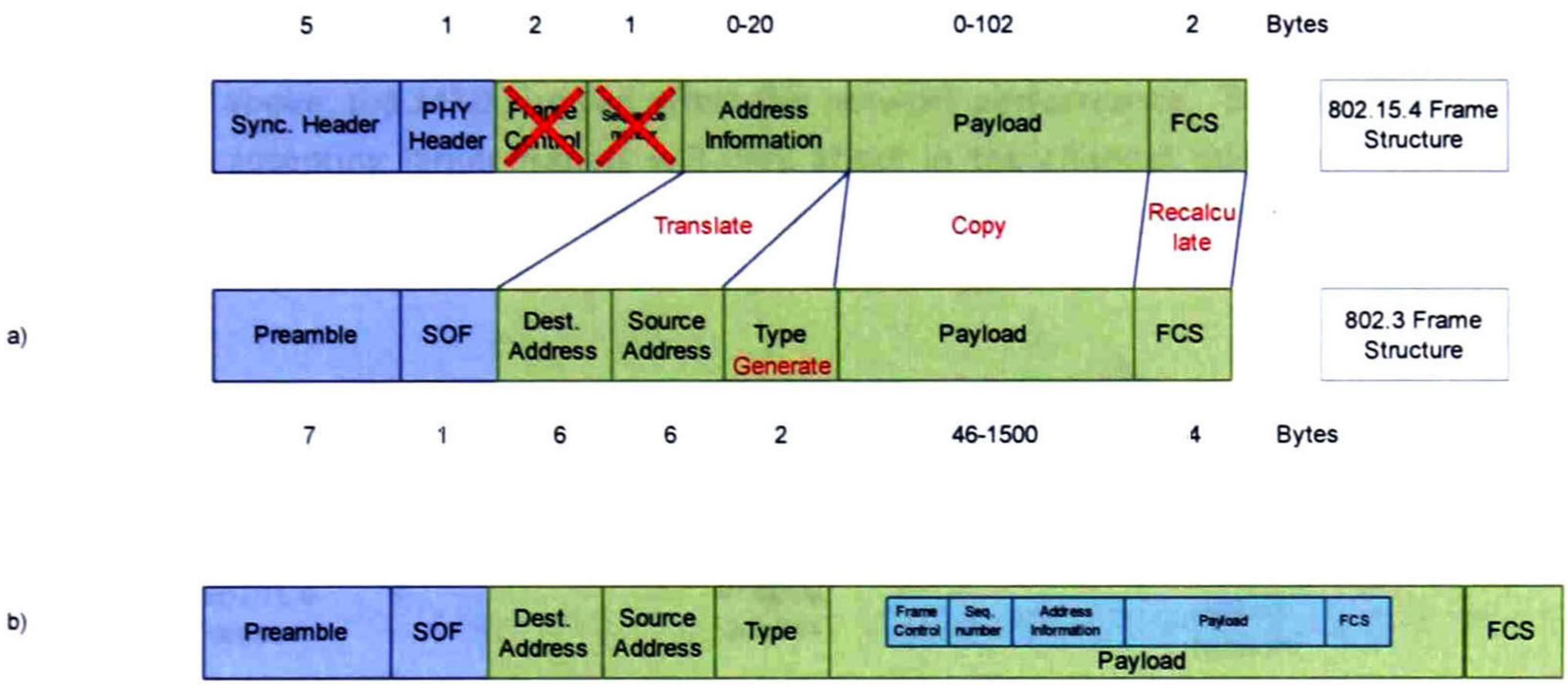


Fig. 6. a) Translation of an 802.15.4 frame to 802.3 frame, b) Encapsulation of 802.15.4 into 802.3 payload

2.5.4 Heterogeneous Bridging Issues

Some pros and cons of two bridging techniques for heterogeneous networks were mentioned. The advantages of one are the disadvantages of the other. There are other issues that affect the bridging process, no matter which bridging technique are used.

2.5.4.1 Maximum Transmission Unit (MTU)

All protocols define the minimum and maximum amount of data that can be transmitted into the payload field. This condition gives some flexibility, but the frame size can affect the entire transmission in different ways, in general larger MTUs gives benefits:

- Increasing the channel efficiency. More user data can be exchanged with the same overhead.
- Reducing the processing overhead. Larger amount of data is sent with fewer frames, reducing the processing in the transmitting and receiving devices.

In the other hand, using larger MTUs affects the network performance:

- Increasing the delay. If one node transmits very long frame, the other stations must wait a longer time.
- Increase the probability of frame loss. A single bit error will cause the frame to be discarded, the more bits in the frame, the higher the probability that one of these bits will be in error.
- Increase the minimum memory requirement. The amount of memory in the receiving nodes must be higher and, in the bridges, the buffers must be larger too.

2.5.4.2 Network specific features

Each transmission technology has unique features and behavior giving some advantage over others. When dissimilar network segments are bridged, it is not possible to extend any network-specific features or behavior across the entire network. E.g. features such as fame size, bandwidth, quality of service, security, etc. can only be provided on the local network segments because the destination network segment will have different MTU or may not support quality of service.

As mentioned above, the MTU size can affect the network performance. Transmitting a little frame to a segment accepting larger frames will only affect in the channel utilization. But transmitting larger frame to a network segment accepting smaller frames will need a frame fragmentation. This situation can increase the delay and processing time (see Figure 7).

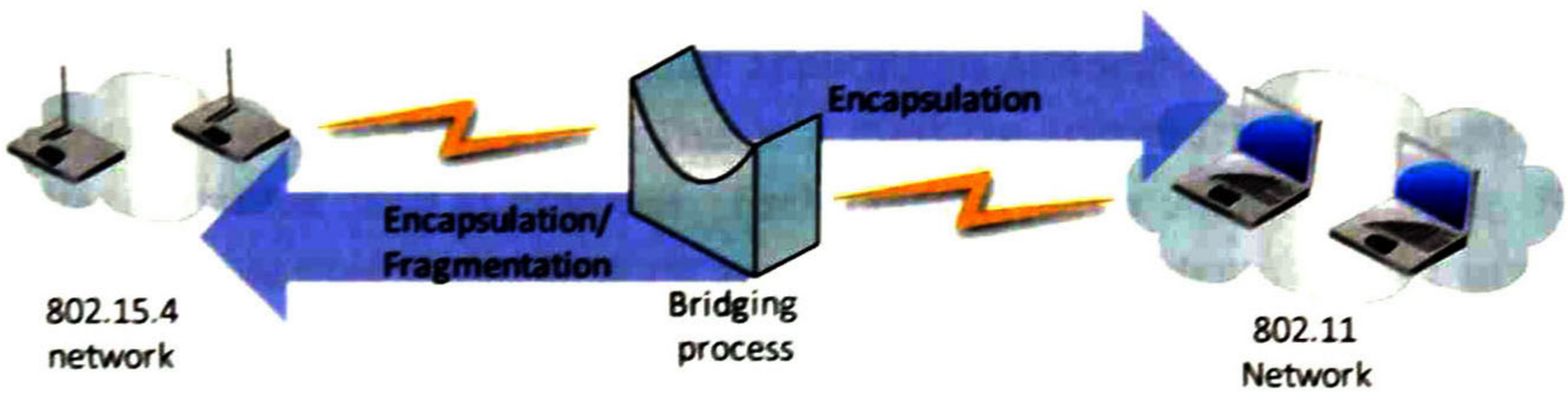


Fig. 7. MTU issues in the heterogeneous bridging process

2.6 Network protocols.

This research will focus in three protocols, one of them used in local area networks: 802.11 (wireless networks); the other two are used generally for sensor networks: Controller Area Network (CAN) used mainly in automotive wired sensor networks and the 802.15.4 (the MAC layer protocol for Zigbee) used in wireless sensor networks. In the following sections the transmission details and the frame structure for each protocol are briefly described.

2.6.1 IEEE 802.11

According to the 802.11 standard [18], if a node wants to transmit, it must first check if the channel is free using the CSMA/CA algorithm. It specifies that the node must have a free channel with a time window equal to the Distributed Inter Frame Space (DIFS) to be allowed to transmit. In the case of a busy channel, the node will delay the transmission until the channel appears free. Once this time has elapsed, the node will select a random backoff time. This counter decreases only when the channel is sensed free and until it gets to a value of 0. When the node transmits the frame, the receiving station will send an ACK frame to notify the successful transmission. It uses a Stop-and-Wait ARQ protocol.

2.6.2 IEEE 802.15.4

According to the 802.15.4 standard [19], a node wishing to send a data frame must use the Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) process periodically and will contend for the channel utilization with the other nodes in the network.

The transmission process starts with a random waiting time chosen in the range of (0, 2^{BE-1}) backoff times (a backoff time is equal to 20 physical layer symbols transmitted where each symbol is composed of 4 bits). Once the initial delay has elapsed, the node will perform a clear channel assessment (CCA). If the channel is busy then the transmission attempt counter (NB) value is increased and a new waiting time is started. If the channel is clear, a second CCA is performed following the same procedure as in the first CCA. If the channel is free in this second assessment, the node starts the transmission. If the node cannot transmit, the node repeats the backoff procedure until the NB counter reaches its maximum value (generally 5). (See Table 1 for the variable meaning).

The receiving node answers only if the reception was successful with an ACK. If the sending node does not receive the ACK frame it assumes failure and retransmits the frame (by default, the sender executes 3 trials). Note that the ACK frame does not use the CSMA/CA procedure

2.6.3 Controller Area Network (CAN)

CAN is a serial protocol designed for vehicular applications although, its use has been extended to industrial automation and domotics [20]. This is a message-oriented protocol unlike most of the protocols address-oriented such as Ethernet. Each message has a unique identifier. This identifier sets the message priority used in the medium access: the lower the identifier the higher the priority. Also the identifier is used to control the collisions in a non-destructive way. If two nodes transmit at same time, the identifier is compared at bit level and the lower identifier is transmitted. The message with low priority is queued to wait for transmission.

The MAC protocol consists in two parts: the contention phase and the transmission process. The contention phase occurs when any initial transmission attempt is made. When two or more nodes are in the contention phase, the ID field of all messages is compared. During arbitration every transmitter compares the level of the bit transmitted with the level that is monitored on the bus. When the bits are different, the node transmitting a bit with value of 1 loses the arbitration (see Fig. 8) and must withdraw without sending any more bits. The transmission procedure is simpler than contention phase: the node continues transmitting the rest of bits, and when the ACK bit is transmitted, the sender waits for the bit modification by the transmitter. This modification acknowledges the transmission. If the modification does not occur in the bit time, the sender assumes that de ACK is not made.

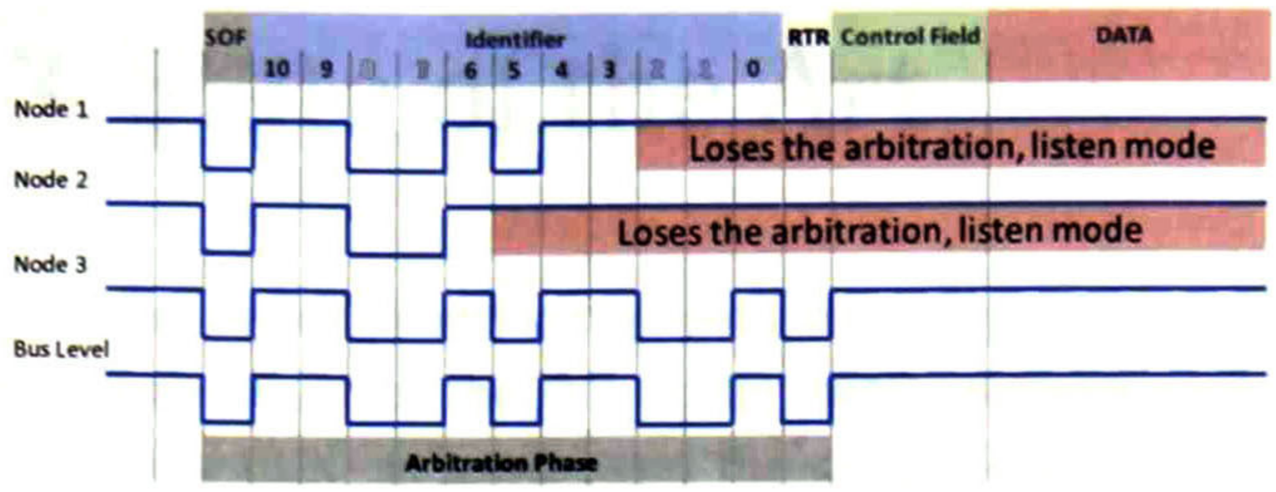


Fig. 8. The CAN contention process.

There are 5 possible errors in transmissions: i) bit error when the value of transmitted bit is different with the sensed value; ii) stuff error when the stuffing rule is violated (after a sequence of five equal bits an inverse bit value must be inserted); iii) CRC error occurs when the CRC calculation of the received message does not match with the CRC field, iv) form errors are detected when the value of fixed bits are different than expected and v) ACK errors occurs when the receiver node does not acknowledges the message. A station detecting an error signals this condition by transmitting an error flag (a sequence of six to twenty dominant or recessive bits) at the next bit time. All stations stops transmitting until all error flag bits are transmitted.

The frame formats including the layer 1 and 2 fields for all reviewed protocols are shown in Figure 9. The numbers under the frame formats represent the amount of bits or bytes needed for each field.

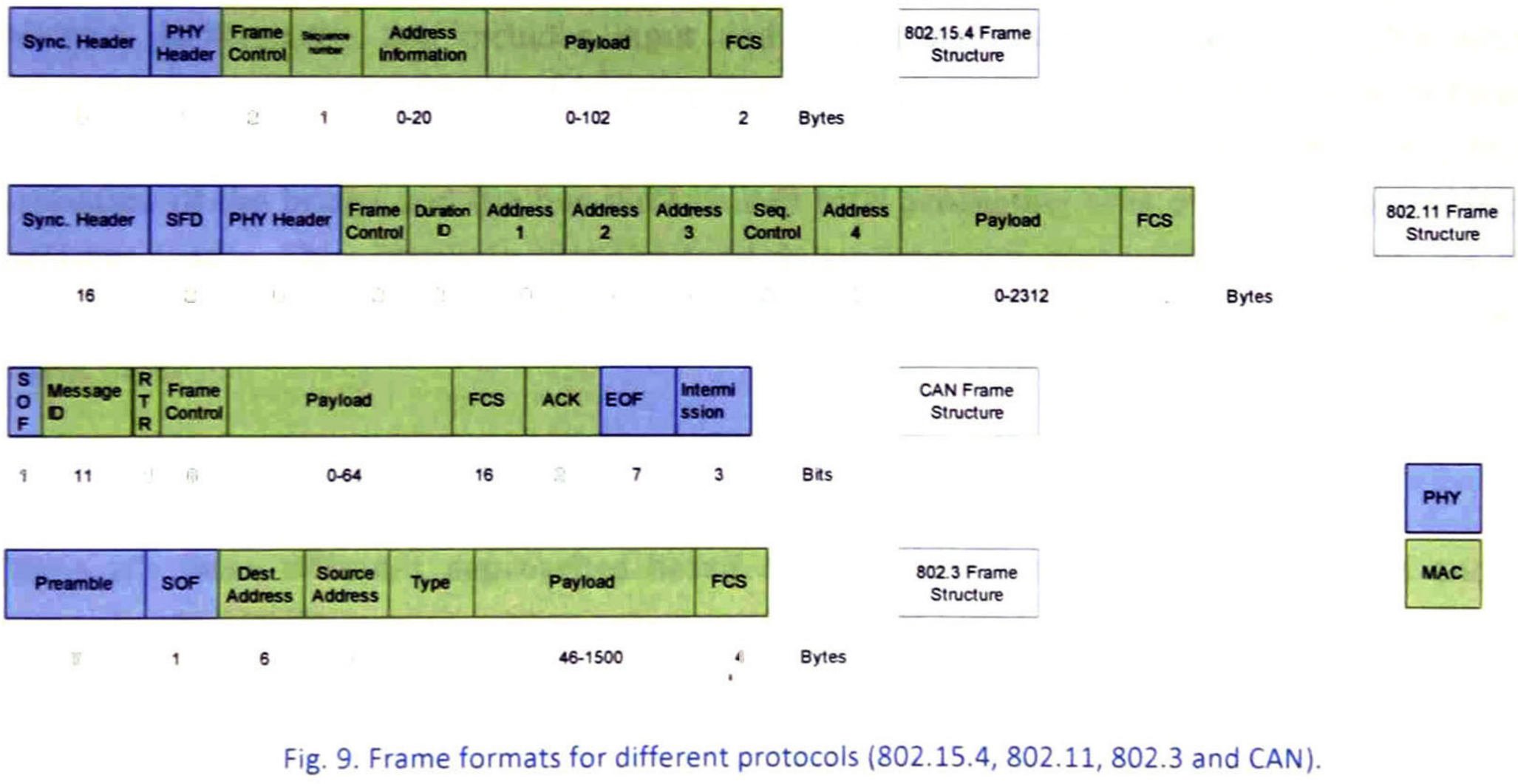


Fig. 9. Frame formats for different protocols (802.15.4, 802.11, 802.3 and CAN).

Chapter 3. State of the Art

There are several approaches to design and implement a network bridge. One of these approaches is to focus the design on one layer of the OSI model (MAC, network or application layer). An alternative approach is to use CLD. In some cases the design follows specific standards such as 802.21 or 802.1D which are focused on the interconnection of network.

3.1 MAC (Layer 2) based bridges

The authors in [21] proposed a bridge to interconnect two CAN segments with an 802.11b WLAN. The bridge makes forwarding decisions using a LT. The retransmission consists in the encapsulation of a CAN message into the payload field of an 802.11b frame. Their simulation results show that with data rates of 100 Kbps and above, the bridge meets the delay requirements of CAN network. With lower data rates, the transmission experiences a delivery delay and increased data loss rate. A similar work is presented in [22] [23] [24] interconnecting two segments of CAN with Wireless ATM, 802.16 and ATM, respectively. Leal et al. [25] proposed a bridge for Wi-Fi and ZigBee using the former as backbone to interconnect the latter domain. It is noted from these proposals that the focus has been mainly to solve the heterogeneity problem for the inter-domain connectivity rather than considering the end to end performance modeling or analysis. The authors in [26] designed and implemented a transparent bridge to interconnect two CAN segments. The bridge has shared memory architecture, and includes input and output buffers to manage the traffic when the processor is processing a frame. A learning process helps to decide if the frame must be forwarded or discarded in the bridge. The performance of the bridge was evaluated empirically from the utilization of the bridge and the bus systems and total processing time of the bridge statistics with different loads. They conclude that the total delay times are acceptable even when the remote messages ratio is high. Also, they found that the minimum buffer size is 5*frame size for the worst traffic case.

3.2 IP (Layer 3) based bridges

There are three different approaches based on IP to interconnect sensor networks and TCP/IP networks: (i) TCP/IP overlay sensor networks; (ii) sensor networks overlay TCP/IP and (iii) bridged based.

In the first approach, the IP protocol is implemented in the sensor nodes. The sensor nodes have an IP address and all nodes use it to communicate with each other (see Fig. 10). The authors in [27] [28] provide a solution to implement the IP protocol stack on sensors nodes which is named **u-IP**. It allows to any node connected to Internet to send information to a sensor node via their IP address. The disadvantage is the processing capacities required and the overhead caused by this implementation. A different approach is used in [29] where tunneling is used to send CAN messages over an IP network. The messages can be transmitted over UDP, UDP/RTP or TCP.

transmissions. This is a matter of discussion of Section 4.3 and part of the main contribution of this Thesis.

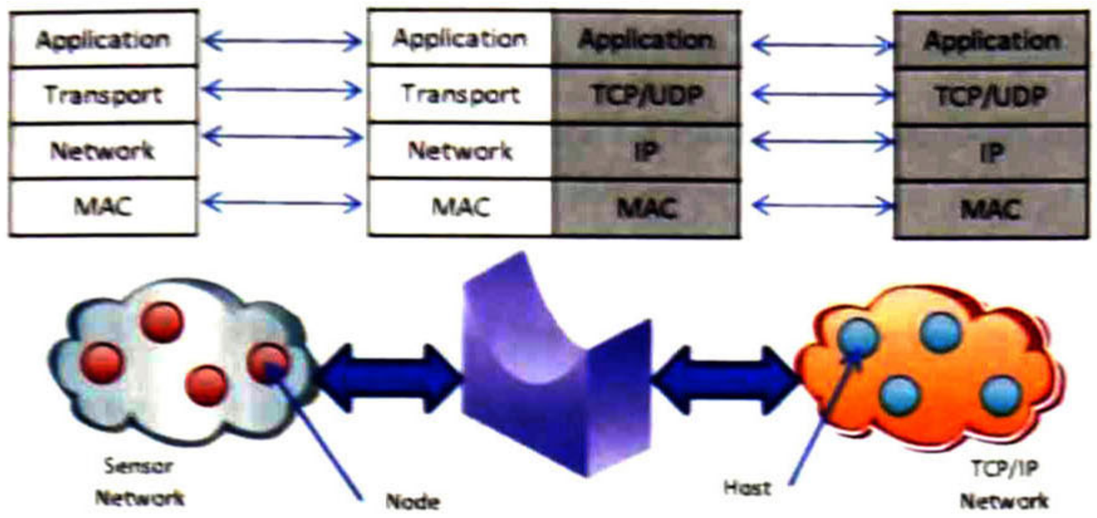


Fig. 12. Interconnection of sensor and TCP/IP networks based on a bridge.

3.3 Middleware (Layer 7) based bridges

A bridge interconnecting two CAN segment with a Bluetooth backbone is presented in [32]. An application from layer 7 encapsulates or extracts the CAN frames into the Bluetooth frame and sends it to the corresponding interface. This bridge does not have a learning process to discard the local frames. The total latency reaches 10 ms. using the higher data rates in both networks, and the jitter is lower than 2 ms.

The authors in [33] propose a hybrid network composed by wireless and wired sensors using ZigBee and CAN respectively. They developed a bridge that interconnects the mentioned two segments and implemented a fragmentation procedure to transmit the ZigBee frames to CAN networks. Using an application based on Smart Distributed System (a CAN application layer developed by Honeywell) and its port to ZigBee network it avoids the problem of translating MAC addresses and node identification. The authors make no performance analysis such as delay or throughput. Also, the bridge does not perform any traffic filter; it only forwards the traffic to one port to other.

3.4 CLD based bridges

Silva et al [34] and El-Azouzi et al [35] studied the interconnection of Wi-Fi and Wi-Max networks based on a CLD approach. The employed bridge is a layer 2 and 3 capable interconnection device. The authors in [36] implemented a bridge based on 802.21 standard (IEEE standard for Media Independent Handover) and conclude that metrics such as packet transmission rate and packet size can be considered irrelevant in the handover time. In [37] the authors show through numerical results that the Wi-Max parameters do not affect the performance of ad-hoc Wi-Fi networks and vice-versa in terms of throughput. On the other hand, the delay is increased affecting the reliability of any delay sensitive services. In these previous works, although end to end analysis is performed, none of them include the bridging process in to their models.

3.5 Specific bridging standards

The authors in [30] propose two bridging mechanisms both based on the 802.1D (IEEE standard for MAC bridges). The first one uses the 802.1Q standard to preserve the QoS characteristics of 802.16 networks and the 802.11e whilst the second mechanism is the same but without support of the

802.1Q standard in order to reduce the implementation complexity. The bridge based on the 802.1Q standard bridge translates the PDUs from the source protocol to the destination protocol frame format. They show that the simulated bridge performs well interconnecting Wi-Fi to Wi-Max; in the other hand, Wi-Fi is observed to behave as a bottleneck for the traffic of the Wi-Max network. The authors propose to verify the Wi-Max data rates to mitigate the bottleneck effect.

3.6 CAN protocol modeling and performance analysis

The performance analysis of the CAN protocol has been addressed in different ways. Davis et al. [41] analyze the response times in nodes and schedulability where a FIFO or priority queues are implemented in their device driver. The results show that when using FIFO queues the required bus speed is increased about 40% and a message priority inversion is detrimental to real-time performance. In [42] the worst case delay is analyzed. The authors use probability distributions to model the number of stuffing bits in CAN messages. They focused on the stuffing bits probability distribution to eliminate the lack of precision of other methods in which the maximum number of stuffing bits is used for calculations. A statistical approach was also used for the same purpose in [43]. The authors compute the probability distributions of CAN messages response times based higher priority message traces. The authors in [44] take a different approach for the delay analysis. They propose a procedure for the detection of the transmission mode for CAN nodes and the evaluation of the message response times using traffic traces. This analysis uses reference messages and their time stamps in the traces for timing estimation. This method flaws when no reference messages can be found in the traffic traces.

The previous works are focused on the worst case delay estimation, but other important metrics such as throughput and loss probability are not addressed. Besides, for [41] [42] and [43] traffic traces have to be generated prior to the estimations.

3.7 802.11 protocol modeling and performance analysis

Bianchi [45] proposed a model of an 802.11 based wireless network. The purpose of the model is to estimate the throughput of such a network when there are *N* active nodes. The author assumes a single collision domain to avoid the complexity of the interference relationship. The model consists in two parts: the device and network parts. The device part models the detailed behavior of the backoff procedure of a CSMA/CA device. The network part uses key performance parameters from the DTMC model to calculate the saturated throughput of the system. As an extension of the previous work the authors in [46] modified the analysis conditions. They show a variation in the scenario in which the non-saturated traffic network is considered and in a subsequent publication [47] the authors added the heterogeneous frame size condition to the analysis. Besides, they introduced variations in the arrival processes to model several traffic patterns, in specific for the VoIP transmissions. In [48] Liu and Lin propose a Markov chain model based on Bianchi´s work. They consider an unsaturated and error prone wireless networks and take into account the packet arrival probability and the frame error probability. They compare the normal acknowledgment procedure versus the block acknowledgment where a node sends multiple back-to-back data frames; each separated by a SIFS period of time and then issues a block acknowledgment request

(BREQ) frame to expect the corresponding block acknowledgment (BACK) frame, obtaining better throughput with their proposal.

3.8 802.15.4 protocol modeling and performance analysis

Mišić et al. [49] modeled the performance of 802.15.4 MAC sublayer using CSMA/CA regime. They take into account several layer-2 factors such as packet arrival rate, number of stations, station's buffer size, packet size and period between beacons. They also derived performance parameters such as the access probability, probability that the medium is idle, queue length distribution in the device and the probability distribution of the packet service time. Using unsaturated traffic conditions, Jung et al [50] modeled the 802.15.4 protocol reflecting several characteristics, including MAC protocol, the superframe structure, acknowledgments and retransmissions to obtain the throughput performance. In their conclusions, the authors mention that they obtain better accuracy for the mathematical model than the Mišić's model. In [51], the authors propose a mathematical model of 802.15.4 using an unslotted CSMA/CA MAC protocol. Using the busy cycle of an M/G/1 queuing system they obtain several performance measures.

Papadimitratos et al. [52] modeled also the 802.15.4 CSMA/CA. Based on a CLD approach, they propose an enhancement to the transmission process by regulating the access to the medium based on the underlying channel quality, taking advantage of existing functionality and signaling to avoid network overhead and achieve simplicity in the implementation. They conclude that with these modifications the goodput increases up to 69%.

In [53] the authors modeled the download procedure in non-beacon mode for 802.15.4 transmissions. The download procedure requires that the end device send a download request frame to the coordinator which answers with an ACK frame and the posterior transmission of the data frame; which is acknowledged by the end device whit another ACK frame. The model includes the CSMA/CA process for both the transmitting and receiving nodes, and the transmission of data and ACK frames. From the Markov chain, they derived the formulae for throughput, delay, loss probability and energy consumption.

Chapter 4 Proposed Bridge Design

In this section the proposed solution is presented in detail which is divided in 4 parts: (i) the processes formalization using process algebra; (ii) the bridge architecture proposal, showing the algorithms and processes involved and the design requirements; (iii) the bridge mathematical model using queuing theory; and (iv) the network models that allow analyzing the respective network segments including the proposed CAN model.

4.1 Bridging Processes Formalization

In this Thesis the studied heterogeneous networks is composed by three different protocols: 802.11, 802.15.4 and CAN.

In order to analyze the performance of the bridge the main bridging processes are identified. The complete operation in a bridge implies four main processes: the reception in each input interface, frame processing for forwarding decisions, lookup table management and frame forwarding in each output interface. All of this processes are independent from each other however, the bridge needs to run all in a parallel way (see equation 1).

$$bridge \stackrel{def}{=} \sum_{i \in port} receiving_i(x) \frown updateLT\langle sa \rangle \frown crossbar\langle proto \rangle \frown \sum_{j \in port} forward_i\langle\rangle \quad (1)$$

Where $x, y = \{sa, da, data\} \cup \{port, ts\}$, $proto = \{802.3, 802.11, 802.15.4, CAN\}$, *sa* stands for source address and *port* for incoming port ID.

Receiving process

When an incoming frame arrives to an input interface or port, the bridge marks the frame with the incoming port ID and a time stamp (equation 2). Afterwards, the bridge applies a filter (equation 3) to discard the local traffic or to place the non-local traffic in the buffer (or queue) corresponding to the input port (5).

$$receiving_{port}(x) \stackrel{def}{=} in(x).(x = x \cup port \cup ts).filter(x) \frown learn(x) \quad (2)$$

$$filter_{port}(x) \stackrel{def}{=} \big(y = searchLT(x{:}da)\big).if\ (y \neq NULL)\ then\ decide(x, y)\ else\ (x{:}port = \text{"}unknown\text{"}) \quad (3)$$

$$decide(x, y) \stackrel{def}{=} if\ (x{:}port \neq y{:}port)\ then\ (x{:}port = y{:}port).\overline{addto - input_{port} - queue}(x)\ else\ drop(x) \quad (4)$$

$$addto - input_{port} - queue(x) \stackrel{def}{=} cont = sizeof\big(iqueue_{port}\big).if\ cont < B\ then\ iqueue_{port}(cont) = x\ else\ drop(x) \quad (5)$$

Where *B* represents the size of $input_{port}$ queue, *da* the destination address and *ts* the time stamp.

Forwarding decision process.

The bridge makes forwarding decision for any incoming frame in each port based on its destination address, see Equation (6). One process extracts a frame from its corresponding input queue (8), analyzes the destination address and places the frame in corresponding output port queue (9). When the destination port is unknown, the frame is broadcasted to all output ports (10).

$$crossbar\langle port\rangle \stackrel{\text{def}}{=} \sum_{p \in port} process(p).crossbar\langle port\rangle \tag{6}$$

$$\begin{aligned} proces(p) &\stackrel{\text{def}}{=} x \\ &= getfrom - input_{port} - queue.if\ (x{:}port = p)\ then\ addto - output_{port} \\ &- queue(x)\ else\ broascast(x) \end{aligned} \tag{7}$$

$$getfrom - input_{port} - queue \stackrel{\text{def}}{=} \left(x = iqueue_{port}(1)\right).\overline{return}(x) \tag{8}$$

$$\begin{aligned} addto - output_{port} - queue(x) &\stackrel{\text{def}}{=} cont = sizeof\left(oqueue_{port}\right).if\ cont \\ &< B\ then\ oqueue_{port}(cont) = x\ else\ drop(x) \end{aligned} \tag{9}$$

$$broadcast(x) \stackrel{\text{def}}{=} \sum_{p \in port} \overline{addto - input_p - queue}(x) \tag{10}$$

Several tasks are performed in the Lookup Table besides its management: search for a destination or source address (11), learning new entries (12-13) (addresses and its incoming port and time stamp), update the timestamps (14-15) and delete the old entries to maintain the LT as short as possible (16).

$$\begin{aligned} search(x, cont) &\stackrel{\text{def}}{=} \left(y = LT(cont)\right).if\ (x = y)then\ \overline{return}(y)\ else\ (cont = cont + 1).if\ cont \\ &= sizeof(LT)\ \overline{return}(null)\ else\ search(x, cont) \end{aligned} \tag{11}$$

$$\begin{aligned} learn(x) &\stackrel{\text{def}}{=} y = searchLT(x{:}oa).if\ (y \\ &\neq NULL)\ then\ updateLTentry(x)\ else\ addtoLT(x) \end{aligned} \tag{12}$$

$$searchLT(x{:}da) \stackrel{\text{def}}{=} cont = 1.search(x, cont)$$

$$addtoLT(x) \stackrel{\text{def}}{=} \left(cont = sizeof(LT)\right).LT(cont) = x \tag{13}$$

$$updateLTentry(x) \stackrel{\text{def}}{=} cont = 1.update(x, cont) \tag{14}$$

$$\begin{aligned} update(x, cont) &\stackrel{\text{def}}{=} \left(y = LT(cont)\right).if\ (x = y)then\ (x{:}ts = presentime)\ else\ (cont \\ &= cont + 1).update(x, cont) \end{aligned} \tag{15}$$

$$\begin{aligned} updateLT\langle s\rangle &\stackrel{\text{def}}{=} wait(secs).\ \sum_{i=1}^{length(s)}\ [get(x_i).if\ (presentime \\ &= ts_i)then\ delLT(x_i)].updateLT\langle s\rangle \end{aligned} \tag{16}$$

From this formalization, it is observed that the bridge processes can affect the entire network (end to end) performance. Also, it is observed that a store and forward based architecture is required for the bridge.

In order to forward the user data through the output ports, header information from the forwarding protocol must be added. This process is called *encapsulation* (see equations (17) (18)). When the received frame payload size is less or equal than forwarding frame payload size, the encapsulation can be used. However, if the data size is less than the forwarding payload field, the throughput will be affected. To overcome this situation it is proposed the use of *frame aggregation* which consists in the encapsulation of two or more frames into a single forwarding frame (19-23).

The opposite case is when the data size is greater than the forwarding frame payload field. In this case, the *frame fragmentation* process is applied which consists in dividing the original data into segments that fit in the forwarding payload (24-27).

Encapsulation Process

$$encap\langle\rangle \stackrel{\text{def}}{=} \left(x = getfrom - output_{port} - queue\right).\overline{out}\left(h_{proto}\right).\overline{out}(x) \quad (17)$$

$$getfrom - output_{port} - queue \stackrel{\text{def}}{=} \left(x = oqueue_{port}(1)\right).\overline{return}(x) \quad (18)$$

Frame Aggregation Process

$$aggregation\langle\rangle \stackrel{\text{def}}{=} \left(x = getfrom - output_{port} - queue\right).aggregation(x, s) \quad (19)$$

$$aggregation\langle v_1, .., v_k\rangle \stackrel{\text{def}}{=} \overline{out}\left(h_{proto}\right).\sum_{i\in I}\overline{out}(v_i).aggregation\langle v_1, .., v_l\rangle \quad (20)$$

$$\begin{aligned} aggregation\langle v_1, .., v_j\rangle \stackrel{\text{def}}{=}\ & \overline{out}\left(h_{proto}\right).\sum_{j\in J}\overline{out}\left(v_j\right).aggregation\langle\rangle + (x \\ & = getfrom - output_{port} \\ & - queue).\overline{out}\left(h_{proto}\right).\sum_{i\in I}\overline{out}\left(v_j\right).aggregation\langle\rangle \end{aligned} \quad (21)$$

$$\begin{aligned} aggregation\langle v_1, .., v_l\rangle \stackrel{\text{def}}{=}\ & if\ l \\ & < I\ \ then\ aggregation\langle v_1, .., v_j\rangle\ else\ aggregation\langle v_1, .., v_k\rangle \\ & + \left(x = getfrom - output_{port} - queue\right).aggregation\langle x, v_1, .., v_k\rangle \end{aligned} \quad (22)$$

$$\begin{aligned} aggregation(x{:}s) \stackrel{\text{def}}{=}\ & if\ |x{:}s| = 0\ then\ aggregation\langle\rangle\ else\ \ if\ |x{:}s| \\ & < I\ aggregation\langle v_1, .., v_j\rangle\ else\ aggregation\langle v_1, .., v_k\rangle \end{aligned} \quad (23)$$

Where h is the forwarding protocol header information, $I = sizeof(WiFiPayload)/sizeof(ZigPayload) = aggregation\ rate$, $1 \leq j < I$ and $I \leq k \leq n$

Frame fragmentation process

$$fragmentation\langle\rangle \stackrel{\text{def}}{=} \left(x = getfrom - output_{port} - queue\right).fragmentation\langle x\rangle \quad (24)$$

$$fragmentation\langle v_1,..,v_k\rangle \triangleq send(v_1).fragmentation\langle v_1,..,v_k\rangle + in(x).fragmentation\langle x:s\rangle \quad (25)$$

$$fragmentation\langle x:s\rangle \triangleq if\ |s| = n\ then\ drop(x).fragmentation\langle v_1,..,v_k\rangle\ else\ fragmentation\langle x,v_1,..,v_k\rangle \quad (26)$$

$$send(x) \triangleq \sum_{i\in I} \overline{out}(h_{proto}).\sum_{j\in J} \overline{out}(byte_{i+j}) \quad (27)$$

Where h is the forwarding protocol header information, $J = sizeof(ZigPayload)$, $I = sizeof(WiFiPayload)/J = fragmentation\ rate$, $k \leq n$ and $byte_i = i^{th}$ byte of WiFi Payload

4.2 Bridge Architecture Design

The proposed architecture for the heterogeneous bridge depicts a device that receives frames on its input ports and forwards them to the appropriate output port and format, based on the destination address (see fig. 11). Considering the previous process description (see section 4.1) the general characteristics for the bridge are described as follows:

The bridge uses the store-and-forward technique: the bridge stores the entire frames before performing any encapsulation, fragmentation or aggregation in the output interfaces.
Transparency: In the inter-network transmissions, the origin node sends information and the destination node receives it. The end nodes not aware that there is an intermediate device.
Translation: in order to forward the frame, the bridge transforms the addresses to the appropriate formats, eliminates unnecessary fields and generates those that do not exist in the source frame, according to the destination protocol specifications.
Multiple input/output queues: this functional characteristic provides flexibility and scalability to the bridge.

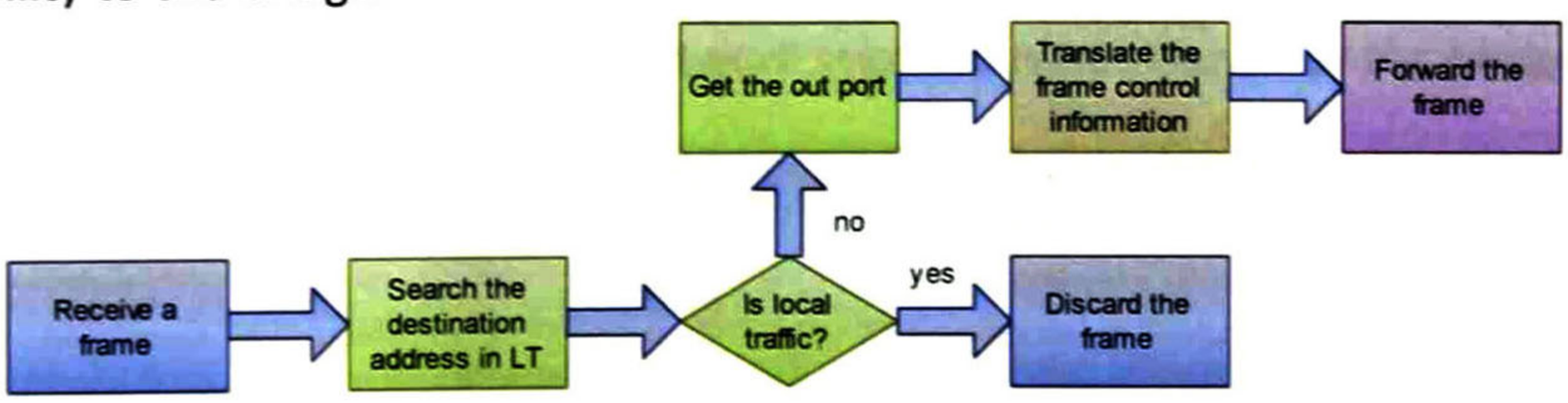


Fig. 13. The general bridging process

The main bridging process depicted in Figure 13 is the base of the proposed design. A detailed view is shown in Figure 14. The incoming traffic filter is responsible for dropping local traffic. The LT stores the source addresses and origin port of incoming frames and this is where the search for the outgoing port for the forwarding frames is performed. The protocol semantics module executes the address translation between protocols. The statistics and reconfiguration modules are responsible for reconfiguring the frame size based on traffic patterns and the channel conditions. The framing modules format the information according to the requirements of the output interface and protocol. The following subsections present further details of the bridge modules..

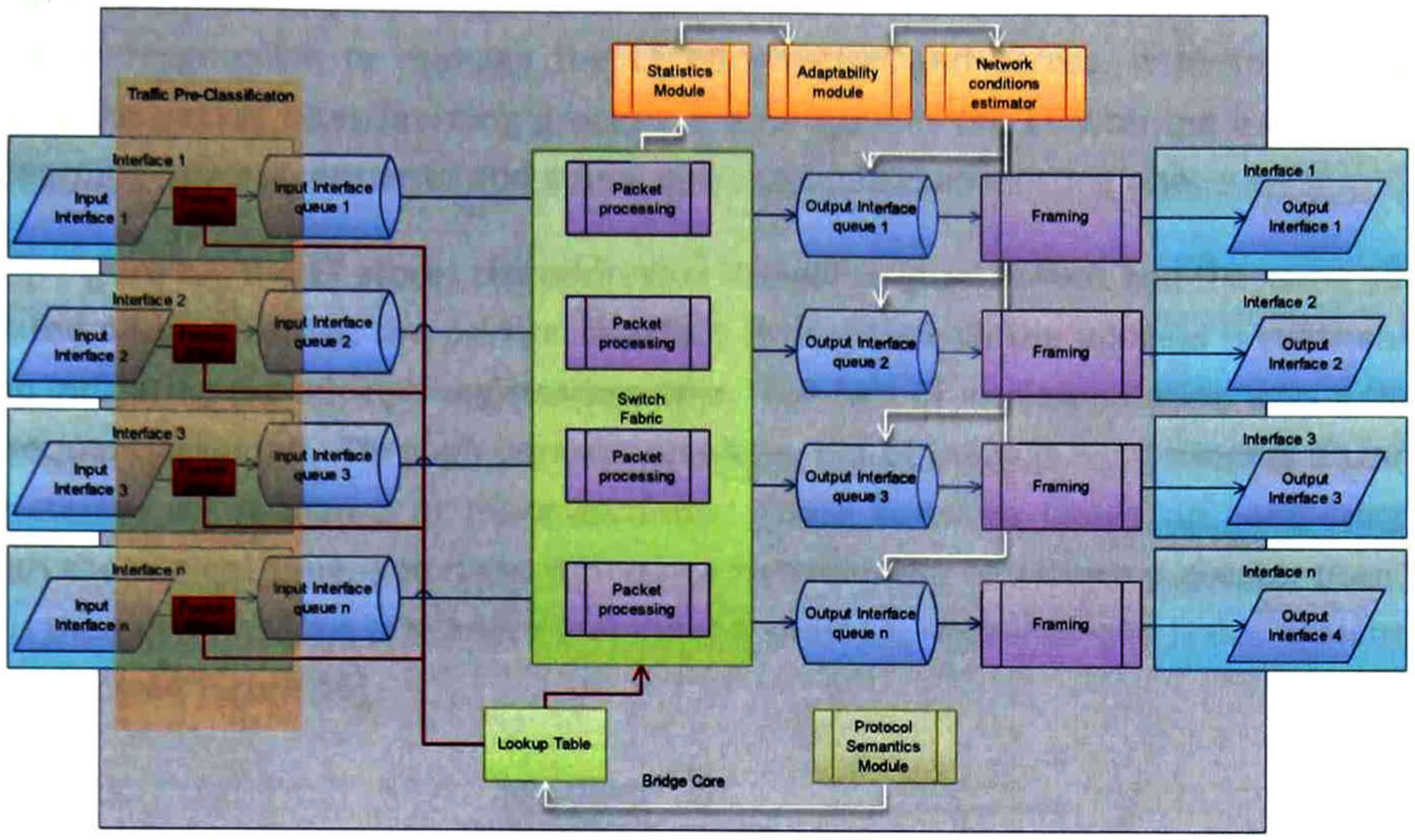


Fig. 14. General architecture

4.2.1 The receiving and address learning process

When a frame arrives to an input interface, the bridge analyzes the frame to get the source and destination addresses. The source address is searched in LT, if the address is not found then a new entry in the LT is made. This entry contains the address, the incoming interface and the time stamp. If the address is found, then the time stamp is updated. With the destination address a second search is made in LT. If the address is found then the source interface of the table entry is compared to the source interface of the frame. When the two interfaces are equal, the frame is discarded (because is local traffic); otherwise the frame is sent to the corresponding output queue. When the destination address is not found in the LT the frame is sent to all output queues (a broadcast to all interfaces), meaning that the node and its network segment is not known by the bridge (see Fig. 15).

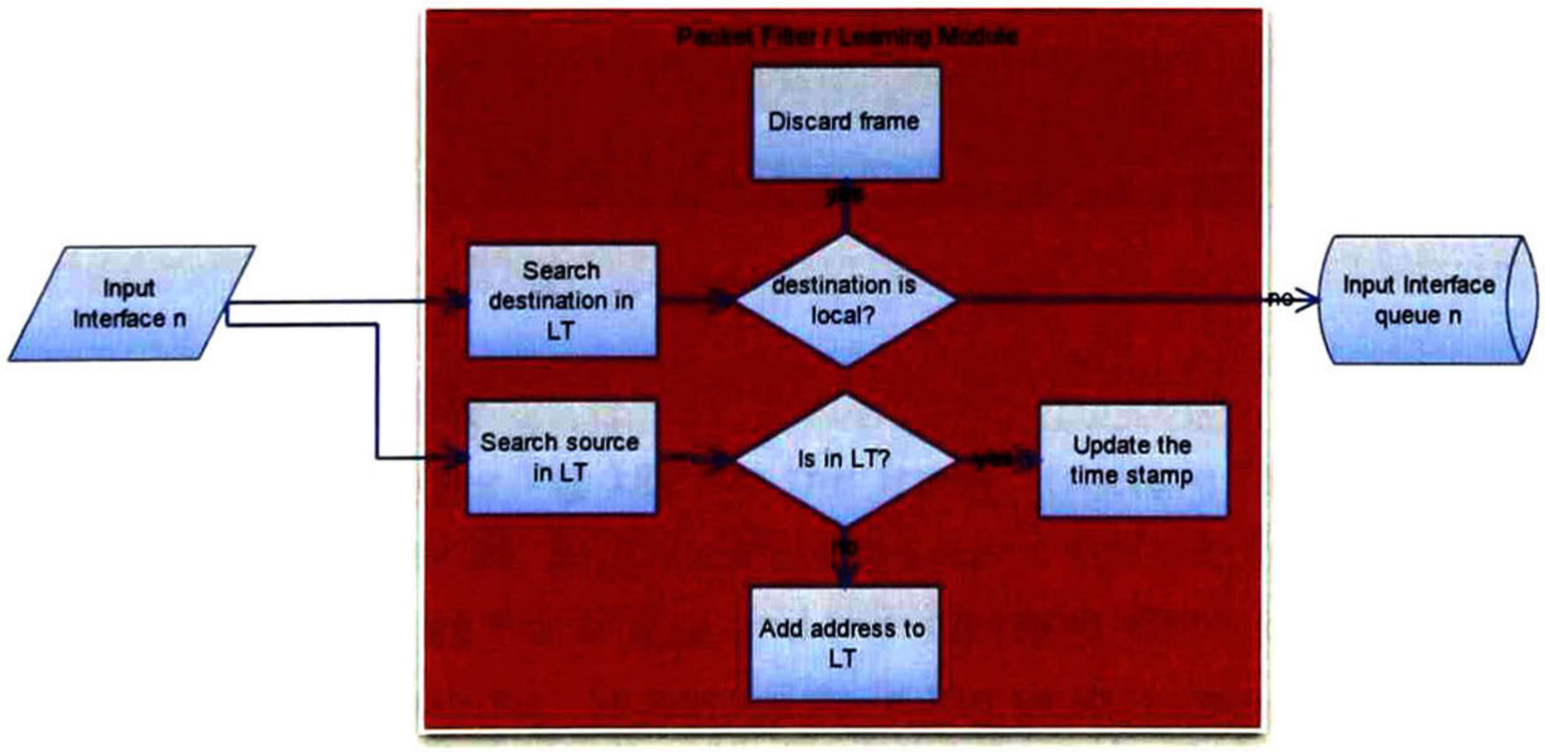


Fig. 15. The receiving and address learning process

4.2.2 Lookup table management process

This process is responsible to manage the LT information and access. It performs the searches requested by the packet filter/learning process. It also updates the LT with the information sent by the filter/learning process, either to add a new entry or update an existing one.

As shown in Figure 17, the LT stores the addresses in their original format and the same addresses in the translated 48 bits format; the port or interface through which the address is reachable and the time stamp indicating the address registration time. The first LT implementation uses a single linked list and a sequential search. Through periodic revisions, the LT management process discards entries with time stamps older than 5 or more seconds. Those revisions consist in comparing the time stamps with the current time, and if the difference between the two times is greater than 5 seconds the entry is then deleted. This procedure maintains a short LT, redounding in faster searches and less memory usage. (see Figure 16)

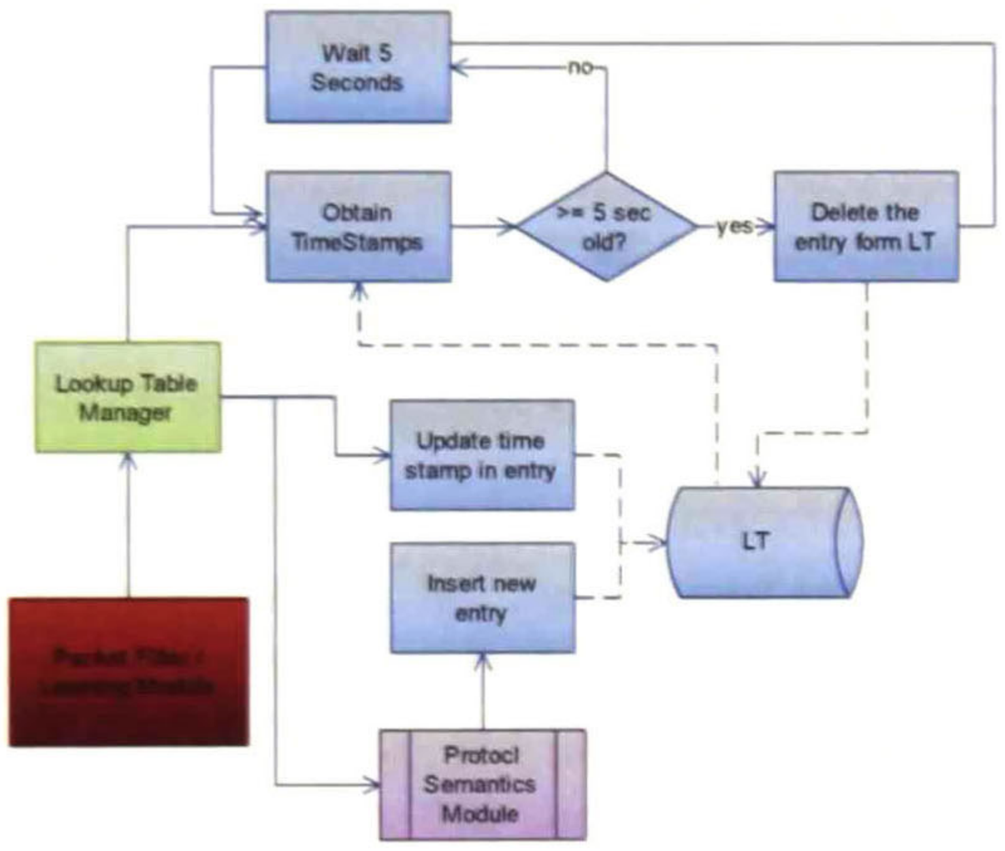


Fig. 16. Lookup Table management process

Original format Address	48 bit format Address	Incoming port	Time Stamp

Fig. 17. Lookup Table format

4.2.3 Protocol semantics module

The protocol semantics module (see Fig 18) has the frame structure for all involved protocols to interpret the frame fields in order to: (i) translate the source and destination addresses from one protocol to any other supported by the bridge and the standard 48 bit address; (ii) recalculate the CRC values; (iii) eliminate the fields that do not exists in the destination protocol; and (iv) simply to copy the contents from the payload field to the destination frame. (See Fig. 6a)

The Table 2 depicts the necessary actions to translate the source addresses from one protocol to any other protocol. Also, the first column is used to translate the addresses to the 48 bits formats.

	802.3	802.11	802.15.4	CAN
802.3	**N/A**	**Use the address as is**	**Take the last 3 bytes and add it to the 0x02 byte**	**Take the last tree nibbles and delete the last bit, convert to binary format.**
802.11	Use the address as is	N/A	Take the last 3 bytes and add it to the 0x02 byte	Take the last tree nibbles and delete the last bit, convert to binary format.
802.15.4	**Add the entire address to the 0x02:00 bytes**	**Add the entire address to the 0x02:00 bytes**	**N/A**	**Take the last tree nibbles and delete the last bit, convert to binary format.**
CAN	Add the entire address to the 0x02:00:00:00:0 bytes	Add the entire address to the 0x02:00:00:00:0 bytes	Add the entire address to the 0x02:00:0 bytes	N/A

Table 2. Protocol addresses translation rules.

In order to homogenize the MAC addresses from all networks and to use that as a key field to perform the searches in the LT, the non-48 bits addresses are translated to the 40 bit format. The 0x02 prefix is used because this is used to indicate a locally administrated MAC address [54].

4.2.4 Packet processing module

This process is divided in two parts (see fig. 19): the first part gets the first frame in the input queue and obtains the destination port by searching in the LT the frame destination address. Once this is done, it constructs the corresponding header and writes the frame information in the output queue which corresponds to the destination port. The second part is the encapsulation process. It looks for the output port framing configuration. This configuration can be: (i) single frame encapsulation where the frame is constructed with the header information and the data from a single frame; (ii) frame aggregation: the output frame is constructed with the header information an two or more (according the output frame size configuration) frames taken from the output queue; and (iii) frame fragmentation where the data is segmented into two or more (according the output frame size configuration) fragments and each frame is sent with the corresponding header information.

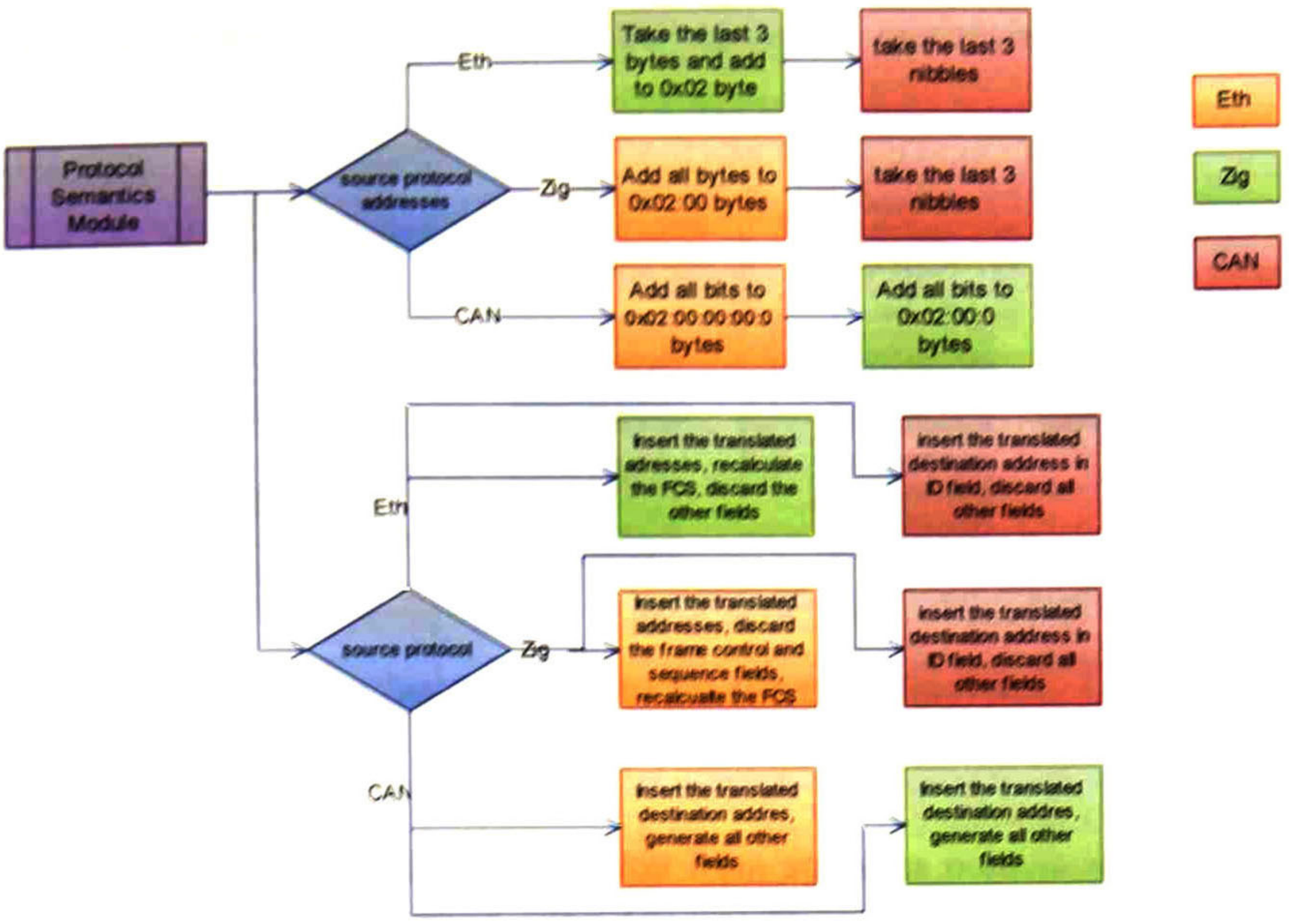


Fig. 18. Protocol Semantics Module

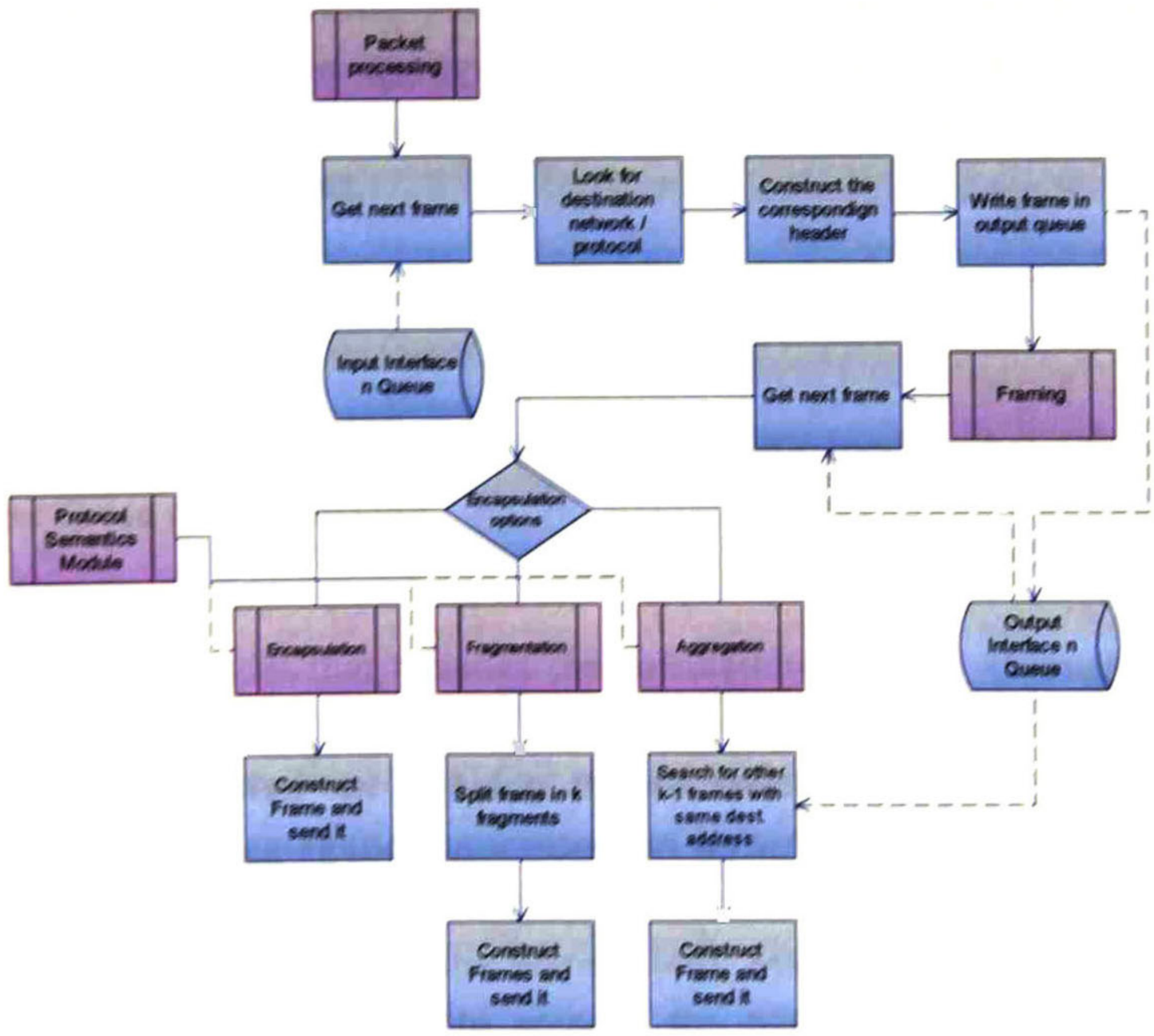


Fig. 19. Packet processing module

4.2.5 Statistics and adaptability module

The objective of the statistics module is to keep track of incoming and outgoing communications of each interface. Counters will be generated for events such as data input and output, errors, frame discards, local traffic and traffic that is forwarded by the bridge. These are expressed in bits, bytes or

frames/sec, and the module will keep track of the minimum, maximum and average indicators of each counter (see fig. 20).

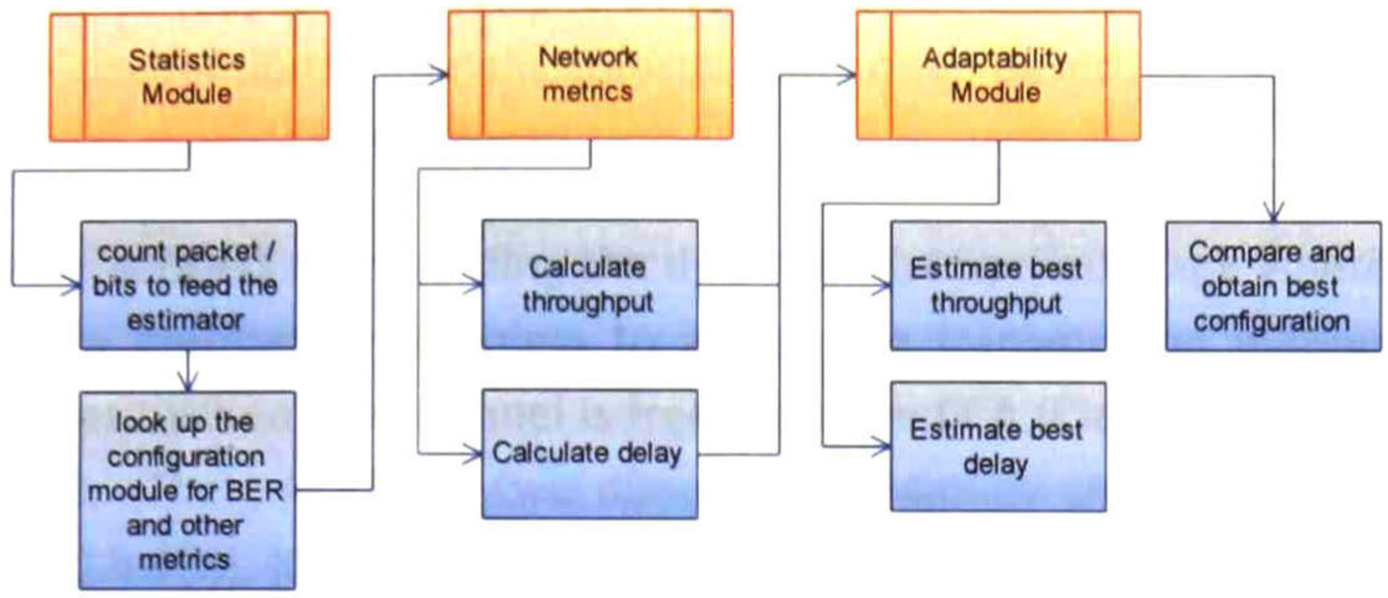


Fig. 20. Statistics, Estimator and Adaptability Modules

The Network Metrics module obtains information from statistics module to calculate the current throughput and delay for each interface. The Adaptability module estimates the best throughput and delay also for each interface using the corresponding mathematical models and information provided by the statistics module. The calculated and estimated value for both network metrics are compared and if the calculated throughput and delay are not near or equal to estimated then a new frame size (and consequently new aggregation and fragmentation rates) will be communicated to bridge the interfaces and the connected nodes. The mathematical models employed by the statistics module are described next.

4.3 Teletraffic Models

This section presents the developed and extended teletraffic models that are proposed as part of the bridge design. These models include:

i. A proposed CAN model that includes the calculation of delay, throughput, loss probability and optimum frame size.
ii. A bridge model based on queuing theory which represents the encapsulation, the aggregation and disaggregation processes (an M/M/1, M/M^m/1 and M^m/M/1, respectively) for CAN, 802.11 and 802.15.4 protocols.
iii. The 802.11 model is based on the Bianchi [45], Malone [46] and Duffy [47] models. This model was extended to take into account the BER from Layer 1 in the delay and loss probability calculations. Also, the throughput (using the BER and ARQ protocol) and best frame size calculations to the model were added.
iv. The same extensions were made for the 802.15.4 model, which is based on the Mišić [49] model. Those extensions where made based on a CLD approach.

4.3.1 The 802.11 teletraffic model

The 802.11 nodes are modeled in a non-saturated traffic situation, i.e. the node doesn't necessarily have a frame to transmit [18] given that the bridge receives the embedded domain traffic at a slower rate than the conventional domain transmission rate. Based on [45], [46] and [47], the conventional domain was modeled as follows:

Let *sw(t)* be a stochastic process representing the delays in the backoff stage where sw(t) ∈ {1 .. M} and *M* is the maximum number of attempts to access the transmission medium, and *bw(t)* the retraction stages counter. When the channel is free after the CCA (Clear Channel Assesment), *bw(t)* will take the value of 0. Then, *W(t)* is a two dimensional Markov chain for the CSMA/CA process with the states ξ={(sw(t),bw(t)), (0,bw(t))e}, and the balance equations shown in (28)

$$P[(0,k)_e|(0,k-1)_e] = 1-q$$

$$P[(0,k-1)|(0,k)_e] = q$$

$$P[(i,k-1)|(i,k)] = 1$$

$$P[(0,k)_e|(i,0)] = \frac{(1-p)(1-q)}{W_0}$$

$$P[(0,k)|(i,0)] = \frac{(1-p)q}{W_0}$$

$$P[(min(i+1,m),k|(i,0)] = \frac{p}{W_{\min(i+1,m)}} \quad (28)$$

Equation (29) shows the probability of at least one station transmitting.

$$\tau = \sum_{i=0}^{m} b_{(i,0)} + b_{(0,0)_e} q(1-p) \quad (29)$$

Let *Dw* be the total delay in the conventional domain derived from the successful transmission delay, the unsuccessful transmission (due to a collision or incorrect reception of the data or ACK frame) delay and the probability of *M* tries to access the medium without any transmission. Then *Dw* is computed as shown in the equation (30). For the throughput estimation the fact that 802.11 use the Stop and Wait ARQ protocol is taken into account. Equation (31) shows the formula for this calculation. In these formulations, *h* is the size of the 802.11 frame header in bits, *R* is the full frame size, *A* is the ACK frame size, and *Ps* is the probability of a successful transmission without collisions and no channel errors, computed as shown in equation (32). Finally, p_e represents the bit error rate.

$$Dw = \sum_{i=0}^{M}\sum_{j=0}^{i}\tau^{j}(1-\tau)^{i-j}\,Ps\left(\sum_{k=0}^{i}W_i - 1 + R + A\right)$$
$$+ \sum_{i=0}^{M}\sum_{j=0}^{i}\tau^{j}(1-\tau)^{i-j}\,(1-Ps)\left(\sum_{k=0}^{i}W_i - 1 + R + A\right)$$
$$+ \sum_{i=1}^{M}\tau^{i}(1-\tau)^{M-i}W_i - 1 \qquad (30)$$

$$Tw = \left(1 - {}^{h}/_{R}\right)\left(\frac{Ps}{Dw}\right) \qquad (31)$$

$$Ps = \tau(1-\tau)^{n-1}(1 - e^{-p_e R_w}) \qquad (32)$$

4.3.2 The 802.15.4 teletraffic model

According to the 802.15.4 standard [19], a node wishing to send a data frame must use the CSMA/CA process periodically and will contend for channel utilization with the other nodes in the network.

The transmission process starts with a random waiting time chosen in the range of (0, 2^{BE-1}) backoff times (a backoff time is equal to 20 physical layer symbols transmitted where each symbol is composed of 4 bits) where *BE* is the backoff exponent, starting with value of 0 and increments after each attempted transmission up to 5. Once the initial delay has elapsed, the node will perform a CCA. If the channel is busy then the transmission attempt counter (*NB*) value is increased and a new waiting time is started. If the channel is clear, a second CCA is performed following the same procedure as in the first CCA. If the channel is free in this second assessment, the node starts the transmission. If the node cannot transmit, the node repeats the backoff procedure until the *NB* counter reaches its maximum value (generally 5). (See Table 3 for the variable meaning).

The receiving node answers only if the reception was successful with an ACK. If the sending node does not receive the ACK frame it assumes failure and retransmits the frame (by default, the sender does 3 trials). Note that the ACK frame does not use the CSMA/CA procedure

For the 802.15.4 network domain analysis a node set $Z = \{z1, z2 \ldots z_N\}$ using the 802.15.4 protocol is assumed. Each node is transmitting to a sink node (which is the embedded part of the gateway) according to a Poisson process with a transmission rate λ.

Var.	Meaning	Var.	Meaning
BE	**Backoff exponent**	**NB**	**Transmission attempt counter**
M	Maximum number of transmission attempts	R	Data frame size in bits
A	**ACK Frame size in bits**	**Wi**	**Random waiting time window**
α	Prob. Of free channel in the first CCA	B	Prob. of free channel in second CCA

Table 3. Meaning of the variables used in the 802.15.4 model

Based on [49]-[52] the transmission process described in the section 1 is defined as follows: let *s(t)* be a stochastic process representing the backoff delays of the CSMA/CA process, where $s(t) \in \{1,\ldots,M\}$. Let *b(t)* be the backoff counter of each *s(t)* transmission attempt where $b(t) \in \{0,\ldots,W_i\}$. When b(t) is in the CCA state it takes the value of -1 and -2 for the first and second CCA respectively. Let *Z(t)* be a two dimension Markov chain representing the CSMA/CA defined as follows:

$$Z(t) = \begin{cases} (s(t),b(t)) & \text{The node is in the backoff procedure.} \\ (s(t), -1) & \text{The first CCA is performed.} \\ \text{The second CCA is performed.} & \\ (-1, k) & \text{The node is transmitting the } k^{th} \text{ bit of the data frame.} \\ (-2, a) & \text{The node is receiving the } a^{th} \text{ bit of the ACK frame.} \end{cases} \quad (s(t), -2)$$

The k^{th} bit of the transmitted frame will be represented by the (-1,*k*) states where $1 \le k \le R$. The ACK frame bits are represented by the (-2,*a*) states where $1 \le a \le A$. Let α and β be the probability that the channel is busy in the first and second CCA respectively. The random delay is chosen in the range of $\{0 \ldots 2^i W_0\}$ where $i = \{1 \ldots M\}$ and $W_0 = 2^{macMinBE}$

If the node has *M* transmission attempts and the cannel is detected busy in both CCAs, the transmission is declared as a failure and the process restarts by default 3 times (maximum 5 times, configurable in the device by the user). When the node accesses the medium successfully, it starts the transmission of the data frame. The frame transmission will be received with no errors with a probability of P_{ACK} in which case the receptor node will transmit an ACK frame. The probability P_{ACK} is calculated using the BER of the channel. The whole transmission success probability is defined as *Ps*.

Let $\pi(i,j) = P[(s(t),b(t)]$ where $i \in \{-1, -2, M\}, j \in \{-1, -2, max\ (R-1, A-1, W_{i-1})\}$, be the steady state probabilities formulated in equation (33):

$$1 = \sum_{i=0}^{M} \sum_{k=0}^{W_i-1} \pi_{i,k} + \sum_{i=0}^{M} \pi_{i,-1} + \sum_{i=0}^{M} \pi_{i,-2} + \sum_{i=0}^{R-1} \pi_{-1,i} + \sum_{i=0}^{A-1} \pi_{-2,i} \qquad (33)$$

Where the first term represents the backoff process steady state probabilities, the second and third terms represent the first and second CCA respectively, the fourth term is representing the data

transmission and the last represents the ACK frame transmission states. The α probability represents the channel in busy state and is formulated as shown in (34); in the other hand, the β conditional probability represents the channel is busy in the second CCA given that the channel was idle in the first CCA, and is formulated ad shown in (35)

$$\alpha = 1 - \left[[R(1-\tau)^{n-1}] \cdot [A(1-\tau)^{n-1}]\right] \tag{34}$$

$$\beta = 1 - \frac{(1-\alpha) - \left((1-\alpha)\tau\right)^{n-1}}{1-\alpha} \tag{35}$$

Where $\tau = \sum_{i=0}^{M} \pi_{i,0}$

The successful transmission probability implies the channel is free in the both CCA and there are not collisions. This is represented in equation (36)

$$Ps = \sum_{i=0}^{M} \pi_{i,0} [(1-\alpha)(1-\beta)] \tag{36}$$

And the probability of failure because the node cannot access the medium in *M* backoff trials is given by (37):

$$Pf = \sum_{i=0}^{M} \pi_{i,0} [\alpha(1-\alpha)\beta] \tag{37}$$

The transmission delay is defined as the successful transmission time in any number of transmission trials, plus the unsuccessful transmission time (caused by a collision or erroneous reception of a data or ACK frame) and the delay caused by the transmission trials without medium access. Equation (38) represents these delays.

$$\begin{aligned} Dz = & \sum_{i=0}^{M} \sum_{j=0}^{i} \alpha^i [(1-\alpha)\beta]^{i-j} (1-\alpha)(1-\beta) Ps(W_i - 1 + Rz + A) \\ & + \sum_{i=0}^{M} \alpha^i [(1-\alpha)\beta]^{i-j} (1-\alpha)(1-b)(1-Ps)(W_i - 1 + Rz + A) \\ & + \sum_{i}^{M} \alpha^i [(1-\alpha)\beta]^{M-i} (W_i - 1) \end{aligned} \tag{38}$$

The throughput is optimized by adjusting the frame size according to the channel conditions. Then, if the frame size (*Rz*) depends on the BER and the channel capacity, the optimal frame size *Rz* can be formulated as:

$$Rz = \sqrt{h/p_e} \tag{39}$$

Let h and p_e be the frame header size expressed in bytes and the BER respectively. The BER formulated in (40) for the specific case of the 802.15.4 protocol using the 2.4 GHz and O-QPSK modulation.

$$p_e = Q(x) = Q\left(\sqrt{2E_b/N_0}\right) \tag{40}$$

Where Q(x) is the co-error function applied to the Signal to Noise Rate formula for 802.15.4 orthogonal signals with E_b as the bit energy and N_0 the noise power spectral energy.

Equation (41) shows the total loss probability P_{loss}. This represents the probability of a failed transmission caused by a collision (the first term), an incorrect reception of an ACK (second term) or a data frame with a bit error (third term).

$$p_{loss} = \left[\sum_{i=0}^{M}[\alpha^i[(1-\alpha)\beta]^i(1-\alpha)(1-\beta)]\right] * (1-Ps) * (1-e^{-p_eRz}) \tag{41}$$

The throughput is calculated by the formula Tz=(1-h)/R which gives the percentage of channel utilization by the payload field. This formula does not consider the Stop and Wait ARQ protocol used in layer 2 of the 802.15.4 standard. The impact of the ARQ protocol is calculated by the division of the probability of a correct transmission (1-P_{loss}) and the total transmission delay. Equation (42) formulates the throughput taking into account the channel utilization and the ARQ protocol.

$$Tz = \left(1 - {}^{h}/_{Rz}\right)\left({}^{(1-p_{loss})}/_{Dz}\right) \tag{42}$$

4.3.3 The CAN teletraffic model

The proposed CAN mathematical model represents the contention phase and the transmission process.

The CAN network is a set T of message transmissions. The CAN network has a set M of messages. Each message $M_i \in M$ in a 4-tuple $\langle A_i, C_i, D_i, V_i\rangle$ where A_i is the {12, 29} bit arbitration field (corresponding to standard or extended ID respectively), C_i is the 10-bit control field, D_i is the {0-8} bytes payload field and V_i is the 18 bit verification field containing the CRC and ACK slots. The transmission $T_i \in T$ is a 3-tuple $\langle P_i, Tc_i, Dc_i\rangle$, where P_i is the priority (given by the CAN arbitration field), Tc_i is the throughput, and Dc_i is the total transmission delay.

The proposed model represents the process of T_i in two phases: i) arbitration and ii) both, control and payload fields transmission. Also, the error frame transmission is considered. The first phase i) is the comparison process of $af \in \{12, 29\}$ bits of A_i in the CAN messages transmitted by a set of $n \in \{2..64\}$ nodes simultaneously. A node enters in this phase with a probability of u_0 and stays in listen state with probability of u_1. The tagged message advances to next comparison state *i+1* (the

next bit comparison) contending with other nodes with probability of λ_i. The node will lose the contention with probability of μ_i. After *af* comparisons, the node will continue to the next phase ii) in the first transmitting state Tx_1 with probability of λ_{af}. The node will advance to next transmitting state with probability of θ_j $0 \leq j \leq 105$. During the transmission of any bit may occur and the node goes to error transmission process with probability of η_i from the contention process and $\eta_{Tx[k]}$ from the transmission process.

The described model is a Markov chain with a state space $\xi = \{(idle)(C_i)(Tx_j)(Err_k)(W)(IFS_l\}$ $where$ $1 \leq i \leq af, 1 \leq j < j'$ $1 \leq k < 20, 1 \leq l \leq 7$. The variable *j'* refers to the sum of the bits corresponding to the control, payload, CRC and ACK fields. The Markov chain has the transition diagram shown in Figure 21, balance equations described in (43) and the meaning of variables in Table 4. The gray circle is the idle state, the blue states set is the contention process, the set of green states is the control, data and verification bits in the transmission process, the light blue circle is the waiting state, the purple states represent the IFS transmission and the orange states set is the error transmission process.

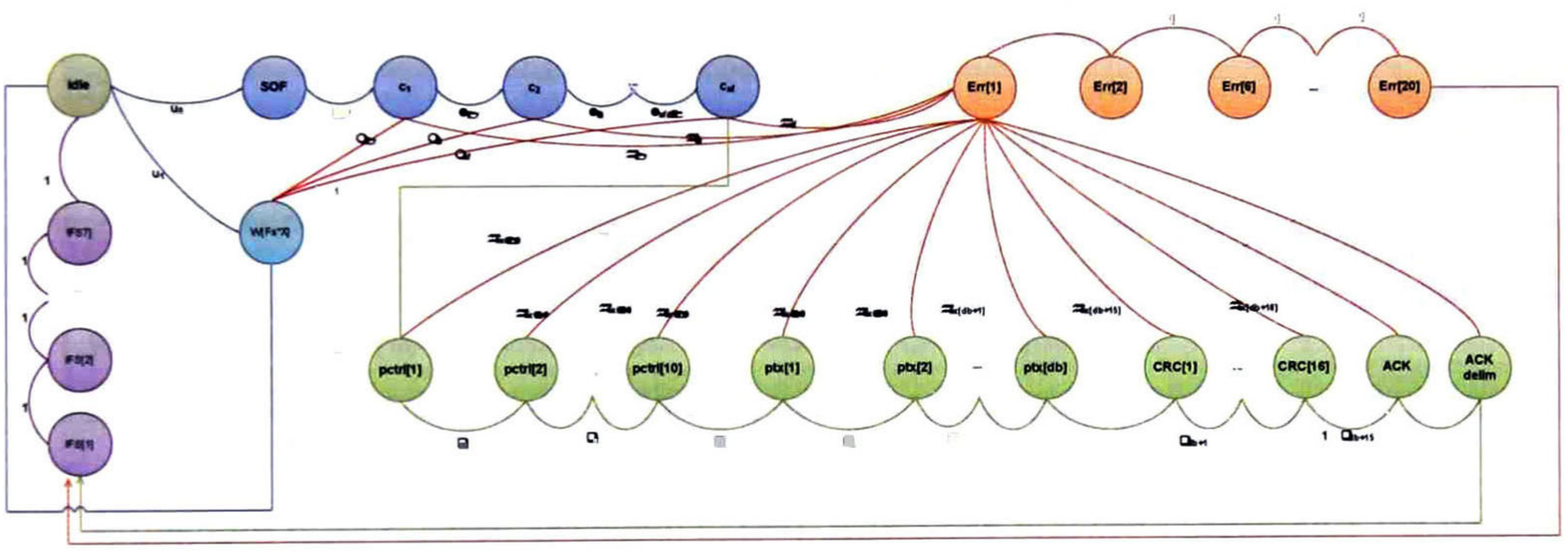


Fig. 21. CAN Markov chain state diagram.

Assuming $\pi = \pi idle + \pi c + \pi Tx + \pi err + \pi ifs + \pi W$ as the vector of steady state probabilities, the balance equations for the described Markov chain are formulated as follows:

$$\pi_{idle} u_0 u_1 = \pi Tx_{j'} \theta_{j'} + \pi err_{20} + \pi IFS_7 + \pi W$$

$$\pi c_1 \lambda_1 \mu_1 = \pi idle u_0$$

$$\pi c_i \lambda_i \mu_i = \pi c_{i-1} \quad 1 < i \leq af$$

$$\pi Tx_1 \theta_1 \eta Tx_1 = \pi c_{af} \lambda_{af}$$

$$\pi Tx_i \theta_i \eta Tx_i = \pi Tx_{i-1} \; 1 < i \leq j'$$

$$\pi err_1 = \sum_{i=1}^{af} \pi c_i \eta_i + \sum_{j=1}^{j'} \pi Tx_j \eta Tx_j$$

$$\pi err_i = \pi err_{i-1} \quad 1 < i \leq 20$$

$$\pi W = \sum_{i=1}^{af} \pi c_i \mu_i + \pi idleu_1 \tag{43}$$

Var.	Meaning	Var.	Meaning
m	**Total number of messages in CAN network**	λ_i	**Probability to check the next i+1 arbitration bit**
n	Total number of CAN nodes contending for the medium in a given time	μ_i	Probability to lose the arbitration in the i^{th} arbitration bit
af	**Bit quantity in the arbitration field**	η_i $\eta_{(k)}$	**The probability that a bit error occurs in the i^{th} bit in arbitration process or in k^{th} bit in the transmission process**
Idle	Idle state, no data to transmit	$\vartheta_{(l)}$	Probability to transmit the l^{th} [control+data+CRC+ACK] frame part
Ci,	**The contention process states**	π	**The vector of steady state probabilities**
$Tx_{(k)}$	The transmission process states	J'	Sum of [control+data+CRC+ACK] bits quantities
$Err_{(l)}$	**The error transmission process**	J''	**Sum of [control+data] bits quantities**
W	The waiting process states	*Rc*	CAN Frame size
τ	**The bit time**	*h*	**CAN Frame header size**

Table 4. Variables used in the CAN model and their meanings

The throughput is formulated as shown in (44), where the first term represents the data portion of CAN frames and second term the ARQ protocol. This metric is the portion of time spent transmitting the payload field, i.e., the channel utilization to send user data. A stop and wait ARQ protocol is assumed, given that the transmitting node waits for confirmation (the bit modification in the ACK slot) from the receiver. If the receiver does not acknowledge the message then the sender tries again starting in the contention phase.

$$\mathrm{Tc} = \left(1 - {}^{\mathrm{h}}/_{\mathrm{Rc}}\right)\left({}^{(1-\mathrm{p_{loss}})\sigma}/_{\mathrm{Dc}}\right) \tag{44}$$

Where $\sigma = Rc/C$ represents the serialization time, C is the channel capacity (in bps) and Rc is the frame size. The loss probability is composed by the probability of errors in the transmitted bits plus the probability of losing the arbitration against messages with higher priority. It is calculated adding the sum of all transition probabilities from c_i to *idle* and Err_1 plus the sum of all transition probabilities from θ_j to Err_1, where $i \leq h$, $j \leq (Rc\text{-}h)$ (See formula 45). It is assumed that an error transmission in a bit occurs independently from others and follows a Poisson distribution.

$$p_{loss} = \sum_{i=1}^{af}(1-\lambda_i) + \sum_{j=1}^{j'}(1-\theta_j) \tag{45}$$

The total delay is as formulated in (46), composed by the time spent by the successful transmission, the unsuccessful transmission because the message was not received or received with errors and the time spent by the node in which it cannot access the medium because of message transmissions with higher priority of size *Rc*.

$$
\begin{aligned}
\mathrm{D_{CAN}} = & \left[\left(\pi idle + \sum_{i=1}^{af} \pi c_i + \sum_{j=1}^{j'} \pi Tx_j + \sum_{k=1}^{7} \pi IFS_k\right)(af + j' + ifs)\tau\right] \\
& + \left[\left(\pi idle + \sum_{i=1}^{Rc}\sum_{j=1}^{i}\left[\pi fr_j * \sum_{k=1}^{20} \pi Err_k\right]\right)(Rc + 20)(\tau)\right] \\
& + \left[2[\pi idle(n * \pi W)(n * Rc)(\tau)]\right]
\end{aligned}
\tag{46}
$$

Where $\pi fr = \pi c + \pi Tx$

4.3.4 Bridge teletraffic model

The bridge's architectural design shows a set of input and output interfaces its respective buffers (see Fig. 14).

The **input interfaces** and their corresponding frame processing are modeled as an M/M/1/B queues. One input interface receives frames from the nodes on a network, either 802.11, 802.15.4 or CAN, hence, the bridge has three input processes, one for each network.

It is assumed that the input processes have a *Poisson* distribution. The frame arrival probability *a* value *(see section 2.2.1)* is set depending on value of the average frame arrival rate λ expressed in frames/sec, according to the formula:

$$a = \lambda/\sigma$$

Where σ is the maximum data rate supported by the analyzed network.

The service times represents the frame processing where the source and destination addresses and the data are extracted from the frames to send it to next bridging process, symbolized by the departure process. It is assumed that the departure times have an exponential distribution. The departure probability *c* value (see section 2.2.1) is set according to the following formula:

$$c = \mu/\sigma$$

Where μ is the average frame departure rate expressed in frames/sec and σ is the maximum data rate supported by the analyzed network.

Hence, the bridge model has three M/M/1/B queues, one for CAN, other for 802.15.4 and the third for 802.11.

The **forwarding decision process** is modeled as a crossbar switch fabric with three inputs and three outputs. The arrival probability for each input is equal to the input queue departure value. As the frames depart from the input queues, arrives to the switch fabric.

The queues for **output interfaces** performing frame *fragmentation* are modeled as an $M^m/M/1/B$ queues and for interfaces performing *frame aggregation* as $M/M^m/1/B$ queues. The input probability for both queue systems is the output probability for the corresponding port in the switch fabric.

The frame aggregation rate *m* is calculated by dividing the size of the forwarding frame payload by the size of forwarded frame payload. This calculation gives the number frames that will be encapsulated in one single forwarding frame.

In the same way, fragmentation rate is calculated by dividing the size of the forwarding frame payload by the size of forwarded frame payload. This calculation lays the number of fragments of the forwarded that will be encapsulated as separated frames.

The **end to end delay** calculation is given by the summation of the transmission delay in all involved network domains and the queuing time in the bridge. This metric is calculated in a *transmission flow basis*. In other words, the end to end delay is calculated for a flow that is originated in one domain, processed in the bridge and then forwarded to a destination domain.

This metric is formulated as follows:

$$D_{ee} = D_{od} + W_{iq} + W_{xb} + W_{oq} + D_{dd} \tag{75}$$

Where D_{ee} stands for end-to-end delay, D_{oo} is for Delay in origin domain, W_{iq} for waiting time in the input buffer on the bridge, W_{xb} is for waiting in the switch fabric (the crossbar switch), W_{oq} for the waiting time in the output buffer in the bridge and finally D_{dd} for the transmission delay in the destination domain.

The **end-to-end throughput** is also modeled in a transmission flow basis. This corresponds to the minimum observed throughput across the transmission in the transmission path and is shown in (76)

$$T_t = \min \ (Th_{od}, Th_{iq}, Th_{xb}, Th_{oq}, Th_{dd}) \tag{76}$$

where the sub-indexes meanings apply as in the end to end delay formulation.

Chapter 5. Analysis and Results

In this chapter, the proposed CAN model validation in presented, comparing estimated performance metric values with the test bed measurements. The model validation was performed for single node and multiple nodes transmitting, varying the payload size in each experiment. Besides, the bridge hardware and software implementation is described, detailing the data flow in the system. In section 5.3, the performance analysis for the heterogeneous bridge is presented. Is presented in two parts, the input interfaces and the output interfaces using either frame aggregation or frame fragmentation.

5.1 CAN Teletraffic Model validation

The proposed model estimations were validated using delay and throughput measurements from the test bed real transmissions. The set up consisted of 5 CAN nodes (ECUs), emulated by the Vector© CANoe software in five stations connected to a physical bus of 40m of length. In all experiments, the nodes transmit at 500 Kbps and the payload size is set from the minimum (0 bytes) to the maximum (8 bytes) values. When the payload size is not specified it is assumed as the maximum value. The first node has 15 different messages, the second has 7, the third has 11, the fourth node has 4 and the fifth node has 6, with transmission periods varying from 10 to 1000 ms. the corresponding IDs and time periods are shown on Table 5.

ID	ECU	Period	ID	ECU	Period
0x0F1	E1	10	0x0C9	E3	12.5
0x120	E1	1000	0x191	E3	12.5
0x12A	E1	100	**0x1A1**	E3	25
0x130	E1	1000	0x3C1	E3	100
0x138	E1	1000	**0x3D1**	E3	100
0x1E1	E1	30	0x3E9	E3	100
0x1F1	E1	100	**0x3F9**	E3	250
0x1F3	E1	20	0x4C1	E3	500
0x3C9	E1	100	**0x4D1**	E3	500
0x3F1	E1	250	0x4F1	E3	1000
0x4E1	E1	1000	**0x772**	E3	1000
0x4E9	E1	1000	0x324	E4	500
0x514	E1	1000	**0x524**	E4	10
0x52A	E1	1000	0x528	E4	10
0x771	E1	1000	**0x77E**	E4	1000
0x0C1	E2	10	0x0F9	E5	12.5
0x0C5	E2	10	**0x199**	E5	12.5
0x17D	E2	100	0x19D	E5	25
0x184	E2	20	**0x1F5**	E5	25
0x1E5	E2	10	0x4C9	E5	500
0x1E9	E2	20	**0x77F**	E5	1000
0x2F9	E2	50			

Table 5. The message priorities and transmission periods per node (ECU)

The first experiment consists in the transmission of a single frame from a CAN source node to a single CAN destination node, varying the payload size from 0 to 8 bytes. We generated traffic traces for the message ID 0x1F3 and measured the delay, throughput and frame. This ID was selected since it has medium priority and this allows observing the delay variations when messages with higher priority are sent. Two cases were studied: (i) when only the 0x1F3 message is transmitted; and (ii) when other nodes concurrently transmit their messages but with higher priority, raising the possibility of collisions at the contention time by overlapping the message periods. Figure 22 shows the delay for the first case (i). The estimated delay is obtained from our proposed analytical model using formula (46) and the measured delay is obtained from the experimental test bed.

Figure 23 shows the variation in the delay when additional nodes start or stop transmitting. In the graphic are marked the moments as t_n {n=1,2, ...}. For example, mark t_1 shows the starting of a second node transmission, ending in t_2. Mark t_3 shows the transmission of two additional nodes and t_6 the transmission using three additional nodes. This graphic data was obtained from measurements in the CAN test bed.

For the second case (ii) all messages in all nodes were transmitted. The measured transmission delays were 0.000254s, 0.000317s, 0.000411s and 0.000497s for only 1, 2, 3 and 4 nodes concurrent transmissions, respectively. Figure 24 shows a comparison between the estimated delays versus the experimental test bed delays.

Figure 25 shows the loss probability. It is observed that with larger payload sizes, this performance metric increases its value. The protocol itself uses many methods to detect and prevent the error occurrence and a very low bit error rate contributes to get very low values for the loss probability. The estimated values where obtained using formula 45 from the proposed model.

In Figure 26, the measured throughput from the test bed is compared with the estimated analytical values from equation 44. The comparison shows alike estimated and experimental throughput values. It is observed that in both the analytical model and the experimental set up the maximum throughput is achieved when a payload size of 8 bytes is employed. The same throughput behavior between our model and the experimental results is also confirmed for all payloads values.

It is observed from the three figures that the proposed model estimations are accurate. The estimated and measured results for all performance metrics are close to being equal. This is confirmed by a correlation coefficient of 0.991, 0.997 and 0.988 for delay, throughput and loss probability, respectively.

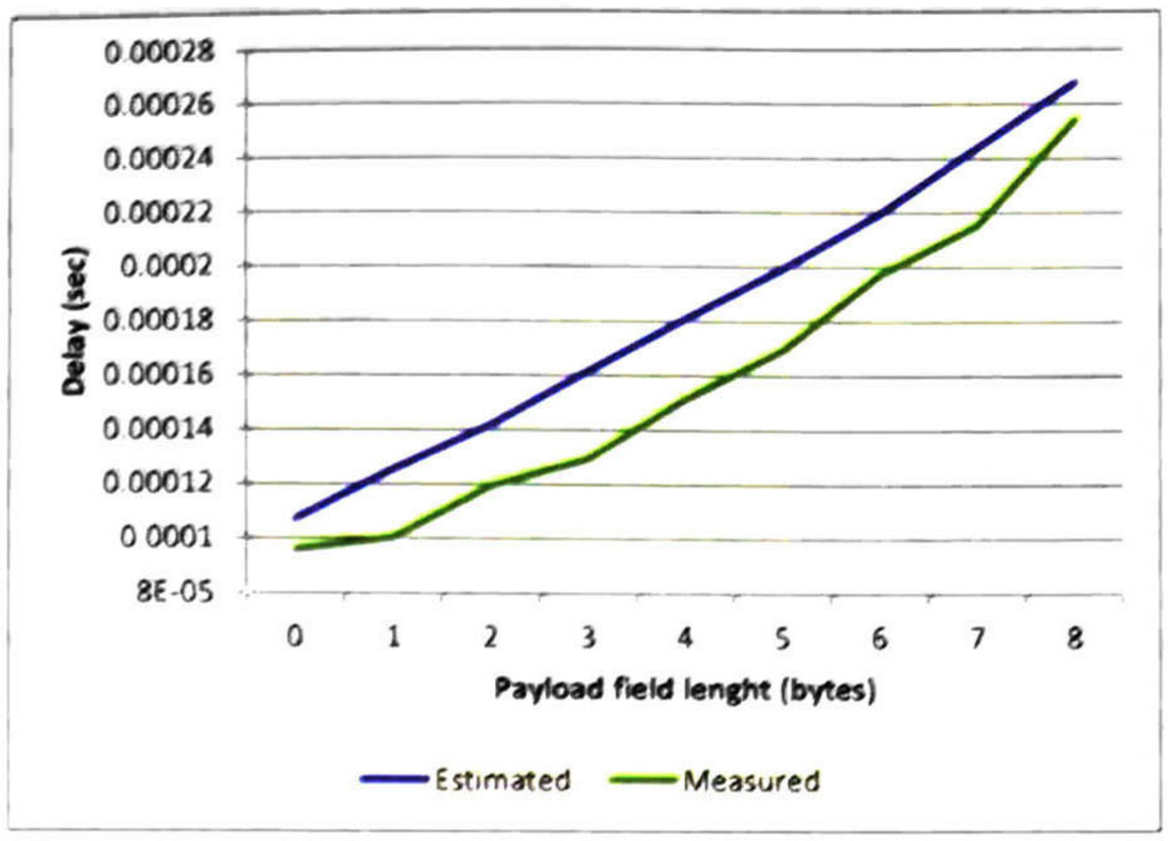


Fig. 22. The estimated and measured delay for message 0x0F1 varying the payload size

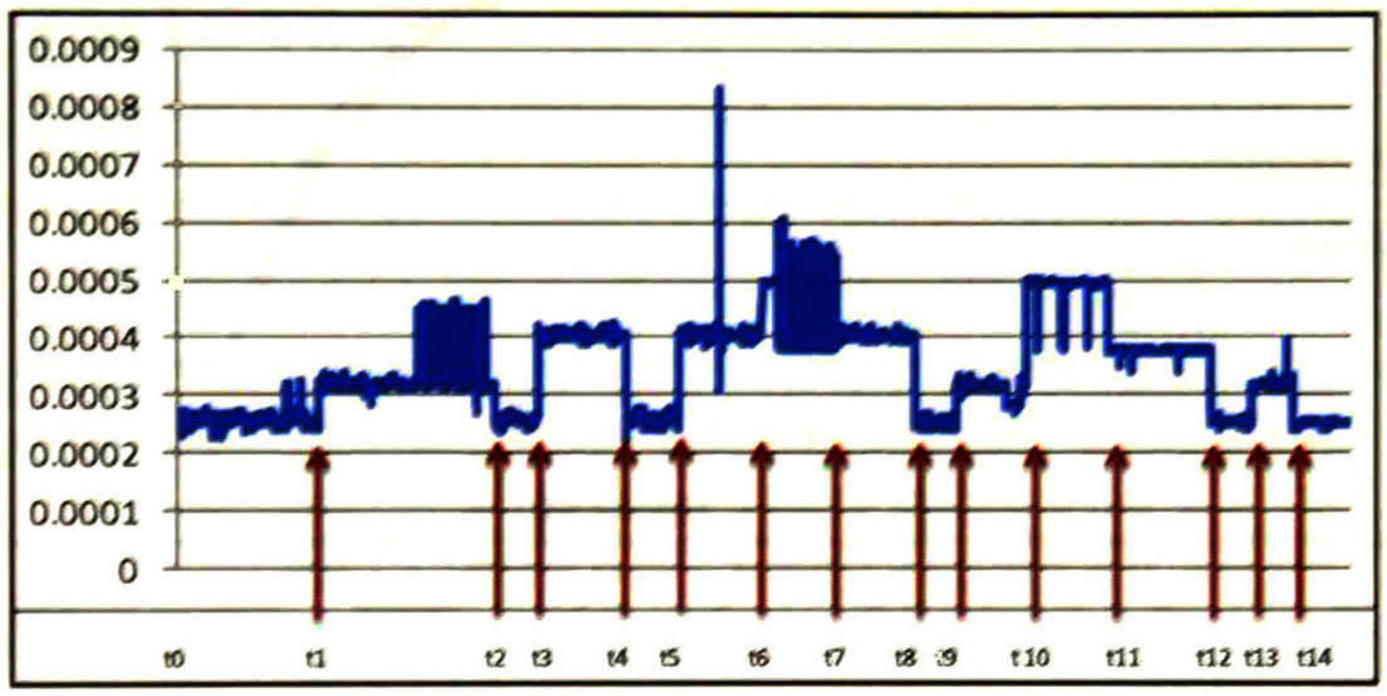


Fig. 23. Delay measurement of message 0x1F3 whit different number of nodes is transmitting simultaneously

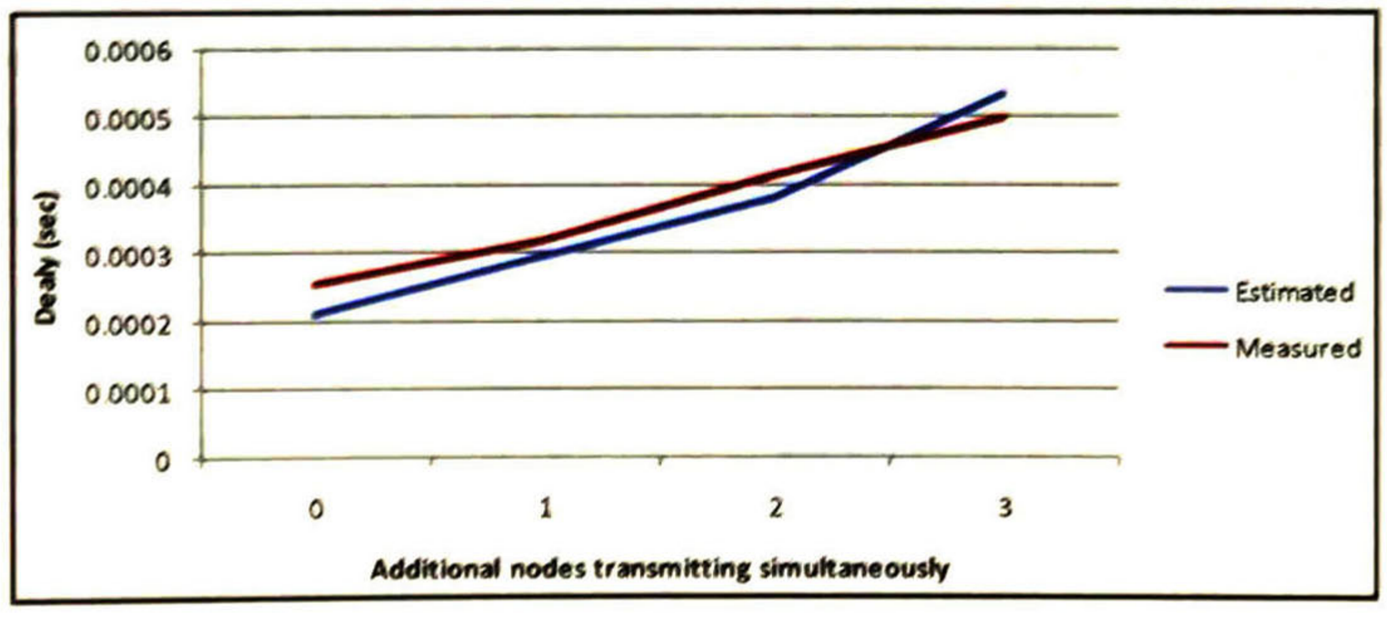


Fig. 24. Comparison between measured and estimated delay for *n* simultaneous transmitting nodes.

The corresponding test bed is formed by a CAN network (the same as in the section 5.1), two interconnecting bridges (CAN – 802.11 and 802.11 – CAN implemented in a Linux system) and two 802.11 nodes which send and receive the traffic. The CAN – 802.11 bridge functions are: i) single CAN frame encapsulation into 802.11 single frame payload; ii) CAN frame aggregation into a single 802.11 frame payload; and iii) network management functions such as monitoring of transmission statistics (time stamps and transmitted frames and bytes per second) and transmission set up (frame size and aggregation rate). In the other hand, the 802.11 – CAN bridge functions are: i) encapsulation of 802.11 frames fragments (a single 802.11 frame is divided into fragments that fit in the CAN payload); and ii) same network management and transmission set up functions as the CAN – 802.11 bridge. The objective of this set up is to obtain the necessary measurements of end-to-end delay and throughput in order to validate the estimated results from our heterogeneous network model.

5.2.1 CAN-802.11 transmissions analysis

Given that varying the CAN frame size (the maximum payload is 8 bytes) effects are minimum in the delay (see Figure 22), we decided to test the end-to-end delay in the bridges using different CAN transmission rates. This was achieved by modifying the periodicity of CAN messages and therefore the frame rate arriving to the bridge. This periodicity changes resulted in frame rates of 405, 900, 1350, 1800, 2250, 2700, 3150, 3600, 4050 and 3500 frames/sec. Finally, when using the maximum frame rate the aggregation rate is changed in the bridge (i.e. the number of CAN frames are encapsulated in a single 802.11 frame) in order to show that when this feature enabled in the bridge the throughput can be increased and the delay reduced.

Figure 28 shows a comparison between estimated versus the measured end-to-end delay. We can observe that the queuing delay within the bridge is minimum compared with the transmission delays in both network segments, besides the 802.11 delay corresponds to the highest portion of the end to end delay. Also, we founded that varying the CAN frame size does not affect the end to end delay significantly. When varying the CAN frame rate, we found that using the highest frame rate the delay is increased only in 0.000036522 seconds in relation to the lowest frame rate (see Fig. 29). From this, we can observe that in the CAN-802.11 direction, the CAN frame rate and size does not affect the end to end delay significantly, being the 802.11 segment which has more transmission delay.

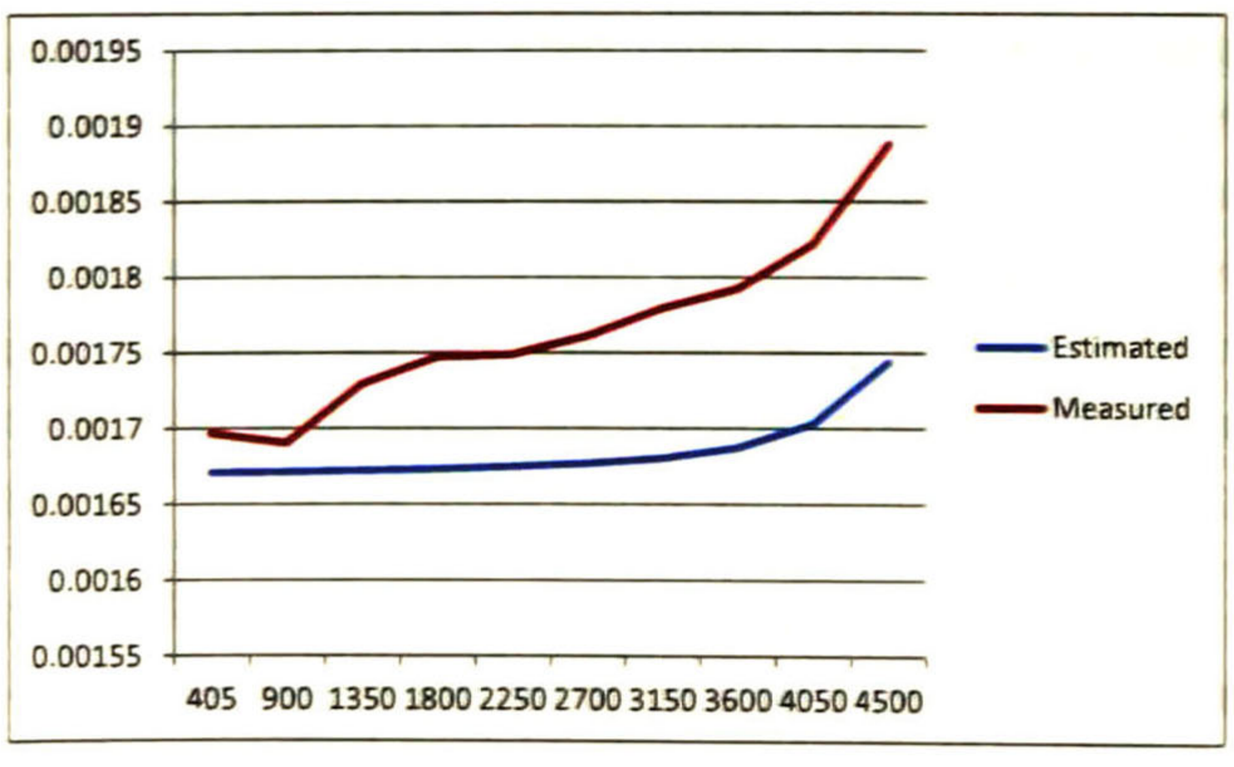


Fig. 28. Estimated and Measured end-to-end delay for different CAN transmission rates

corresponds to the lowest experimental 802.11 frame rate. Frame loss is considerably affected as the traffic load increases in the network. The lowest frame loss is presented for a frame rate of 4 frames/sec, while the highest for 110 frames/sec. Based on the results we conclude that the lower the throughput in the bridge the higher is the frame loss. We note that the queuing system (bridge) becomes unstable for frame rate values higher than 103 frames/sec.

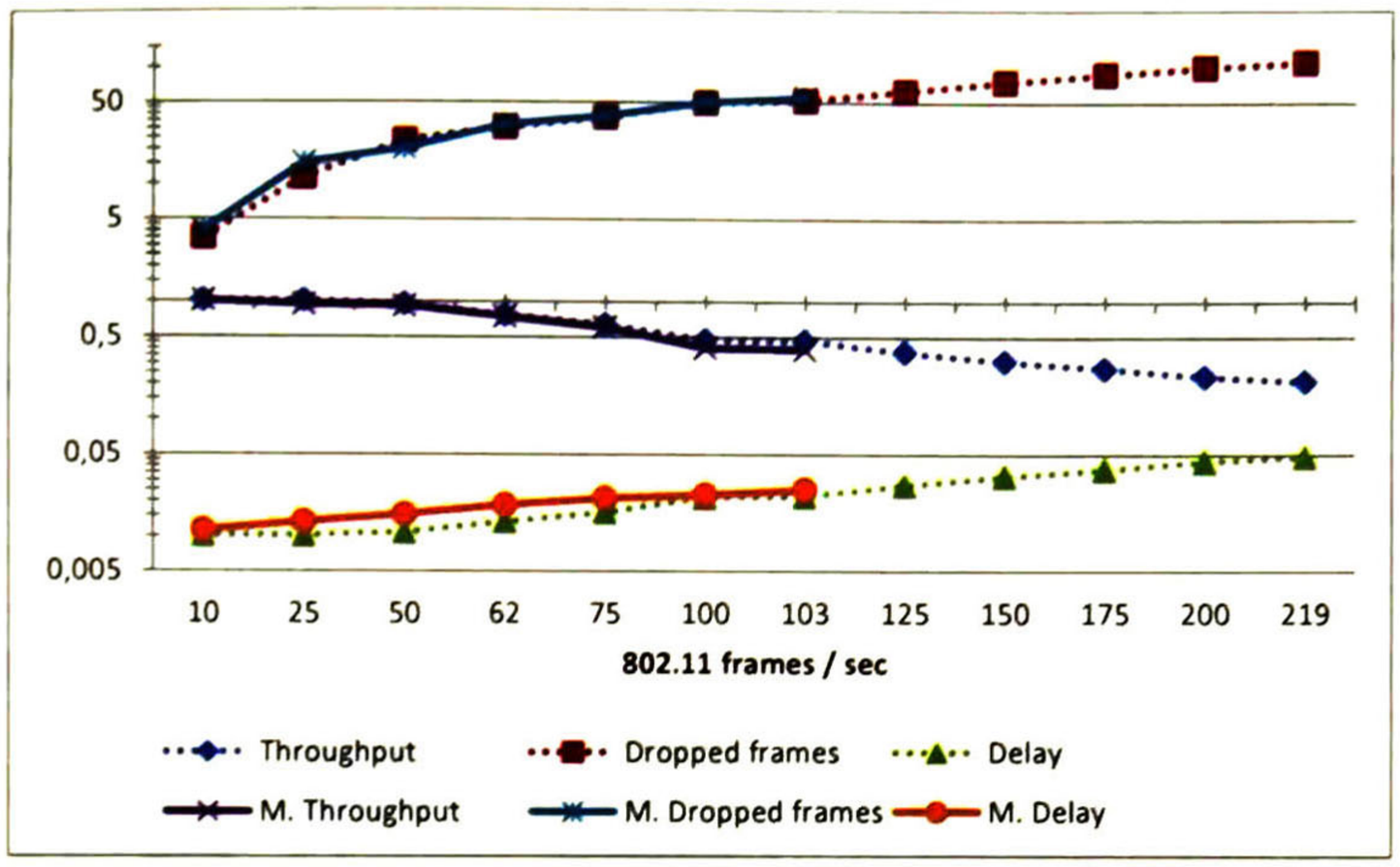


Fig. 30. Estimated and measured performance metrics in 802.11 – CAN bridge.

5.3 Prototype of Heterogeneous Bridge implementation

In order to verify the bridge proposal presented in section 4.2, a prototype was implemented according to these specifications. The prototype was implemented using a Linux operating system on a computer with an Intel™ Atom N270 processor @1.6 GHz and 1GB in RAM. The prototype was built in standard C language (no object oriented).

As 802.11 interface for the bridge the built-in Wi-Fi network card was used (a Realtek™ RTL8187B NIC). The bridge receives frames from the 802.11 network obtaining the full frame (source and destination address information and the user data field) using a *raw socket* from the *inet* C library. In the same way the information is sent to the network nodes constructing the frame as a string of bytes for the raw socket. Using the same library the interface is set to *promiscuous mode* which allows listening to all network traffic. In all cases the MTU is configured using the same software library.

For the CAN interface the bridge employs a GridConnect™ PCAN USB adapter. The Linux device driver allows listening the network traffic and full frame information access (frame ID, user data, DLC and CRC fields).

The 802.15.4 protocol was implemented using a Digi™ XBee Pro S1 module mounted on a USB/serial converter. Full frame access is enabled (the source address, the AT command, user data and CRC field) in the serial interface by setting the XBee module to API mode. The software for the bridge is a multithread implementation. There was one thread for each input and output interface, another thread to manage the lookup table and the main thread which controls the other.

All queues are implemented as a linked list with fixed size; the M/M/1/B takes the last element of the list and sends it out to the next bridging process (switching fabric or forwarding process). The $M/M^m/1/B$ queue waits until *m* frames have arrived and then assembles these frames as one string of bytes before transmission. In the other hand, the $M^m/M/1/B$ divides the data field into *m* fragments sending each one as a single string of bytes to the output interface.

Figure 31 shows the data flow between the software components and the hardware interfaces.

For the current bridge software implementation the configuration parameters must be modified at codification time and changed according to the experimental scenario. The adaptability module implementation (the orange boxes in Figure 14) is not part of the scope of this thesis. This is considered as future work.

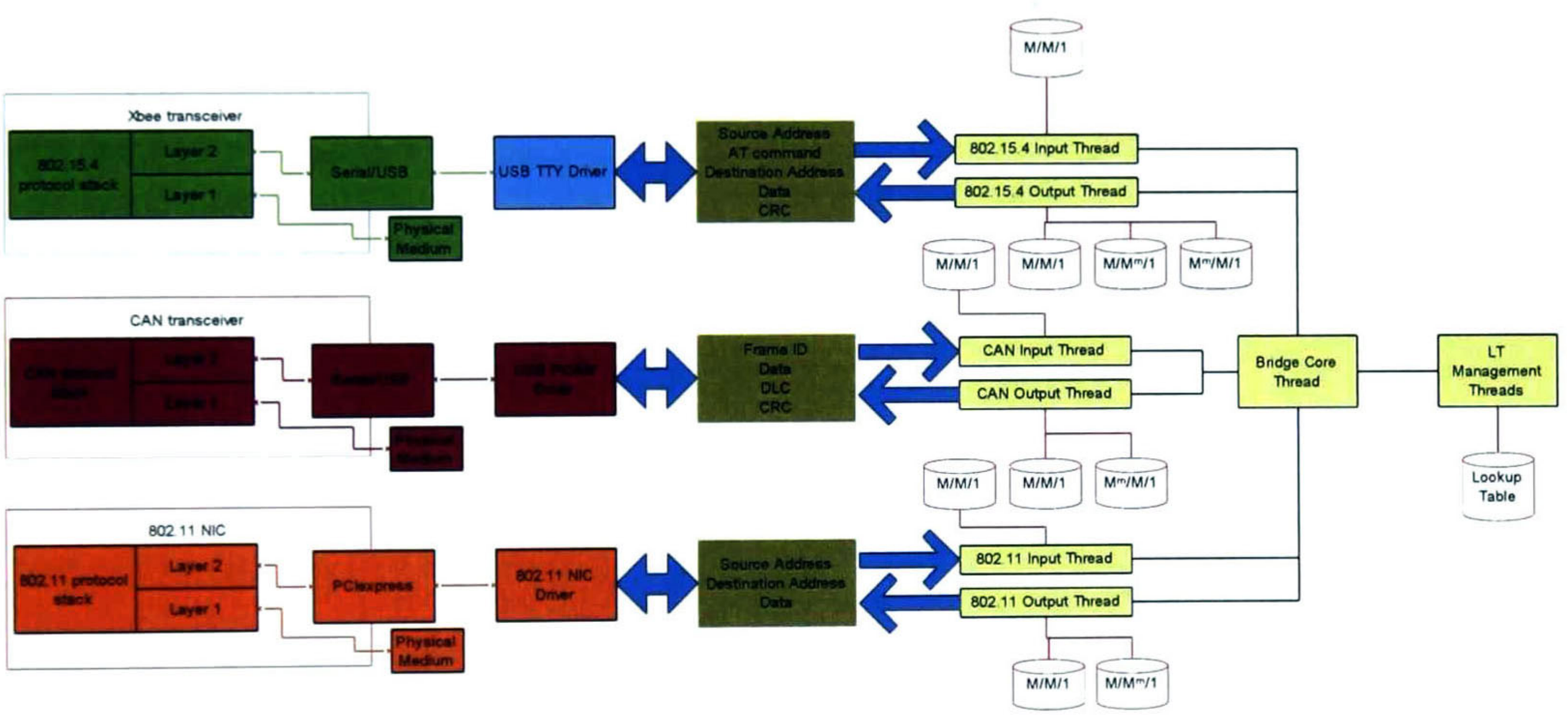


Fig. 31. Data flow in the prototype bridge implementation

5.4 Bridge performance analysis

The performance of the heterogeneous bridge is analyzed in this section. As described in Chapter 4, the bridge architecture is based on a set of buffers for both the input and the output interfaces and a switching fabric. Different performance metrics are employed for these bridge elements in order to derive the overall performance. This includes delay, throughput and frame loss probability. Based on this analysis it is possible to derive the bridge configuration parameters (frame and buffer size, frame rate, etc.) that achieves the best performance for any given scenario. The studied protocols includes 802.11, 802.15.4 and CAN.

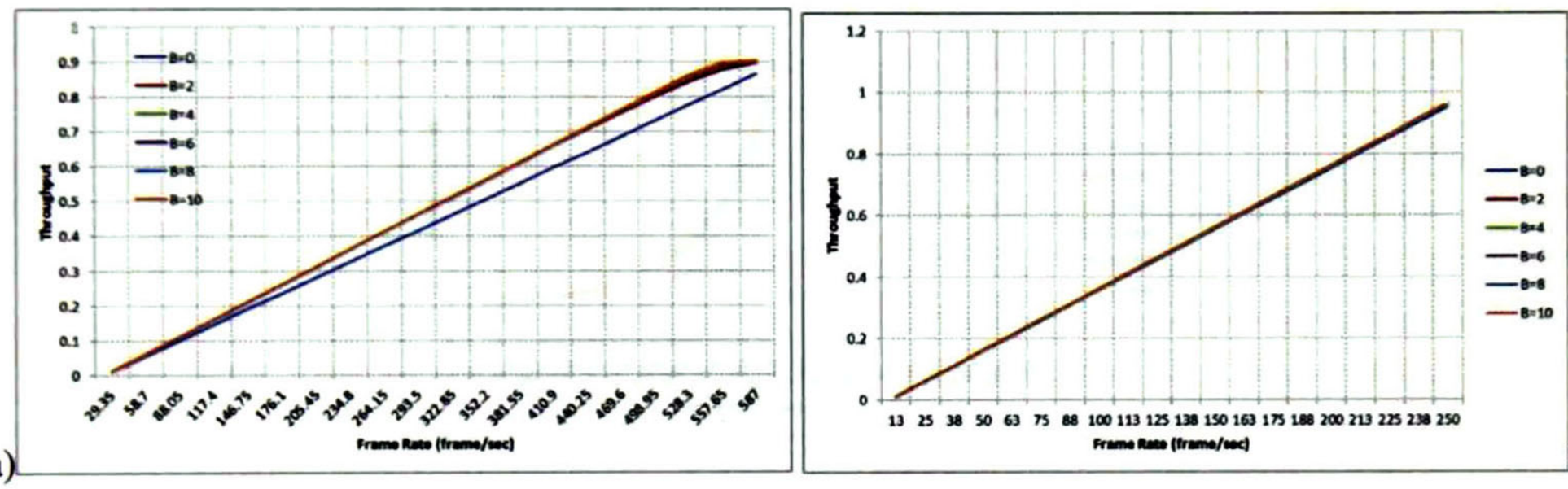
5.4.1 Input buffer analysis

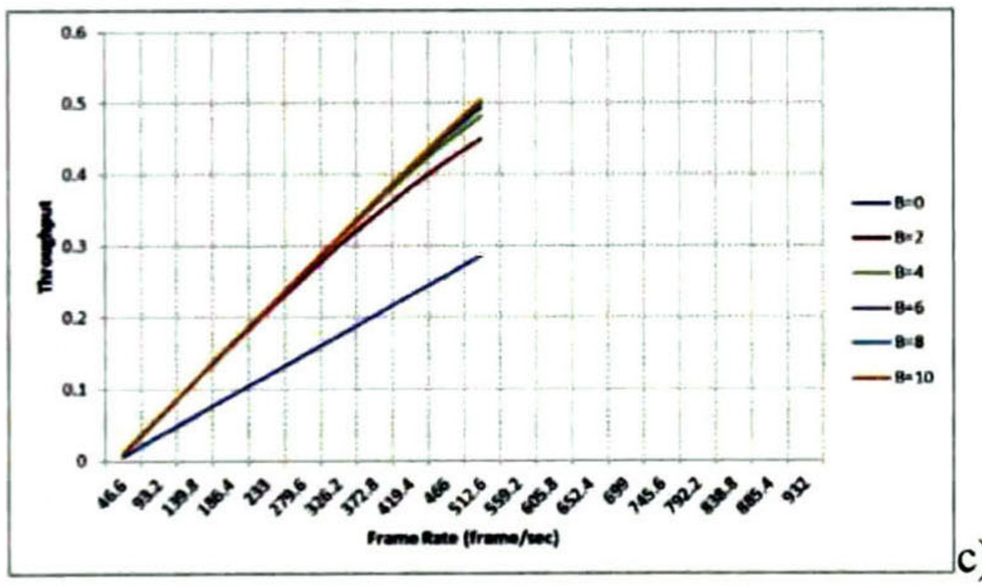
This subsection addresses the performance analysis of input buffer. For all experiments the transmitting stations were configured to use the maximum frame size. The frame rate was varied from 1% to 100 % of channel capacity at different increments and according to the studied protocol. Different buffer size buffer sizes were configured in the bridge (0, 2, 4, 6, 8 and 10).

5.4.1.1 Throughput analysis

Figure 32 shows the throughput for three input buffers, one for each studied protocol. It is observed that as the arrival frame rate increases the throughput increases. For the 802.11 segment the maximum achieved throughput was about 90%, as shown in Figure 32a. For the 802.15.4 and CAN interfaces the maximum observed values were 96% and 50%, respectively.

The current bridge prototype uses multiple threads to control individually each process as described in Chapter 4. For the hardware and software implementation the maximum frame rate achieved was close 500 frames per second for the 802.11 and CAN interfaces. This condition limited the experimental scenarios and it explains the maximum CAN interface throughput of 50% from Figure 32c.





c)

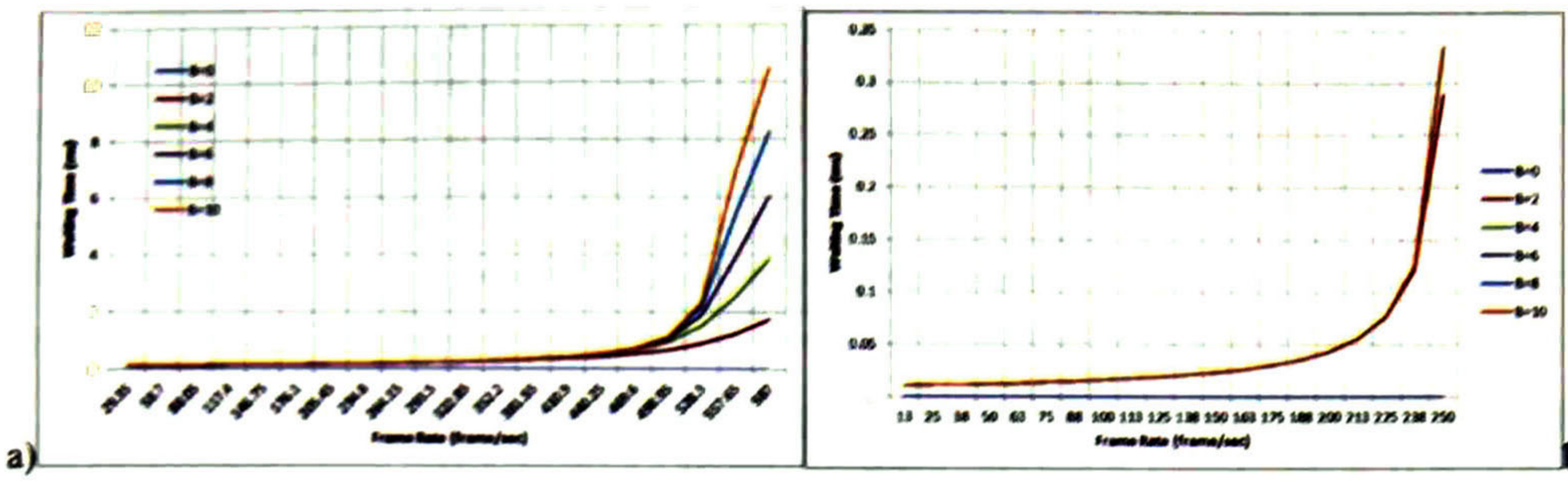
Fig. 32. Throughput for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces

From the previous figure it is observed that the throughput is not significantly affected by the buffer size changes. In other words, it is possible to establish the buffer size to 1 or higher number with a minimum effect on this metric.

5.4.1.2 Delay analysis

In Figure 33 presents the delay for the input interfaces. For the 802.11 interface, the delay remains lower than 1ms until the arrival rate reaches the 500 frames/sec (see Figure 33.a). In Figure 33.b the delay suffers minimum increments because the buffer capacity to forward frames is sufficient for all possible 802.15.4 frame rates. As shown in Figure 33c the delay for CAN interface raises dramatically

as the frame rate is increased. This can be explained by the queuing time derived from the employed frame rates.



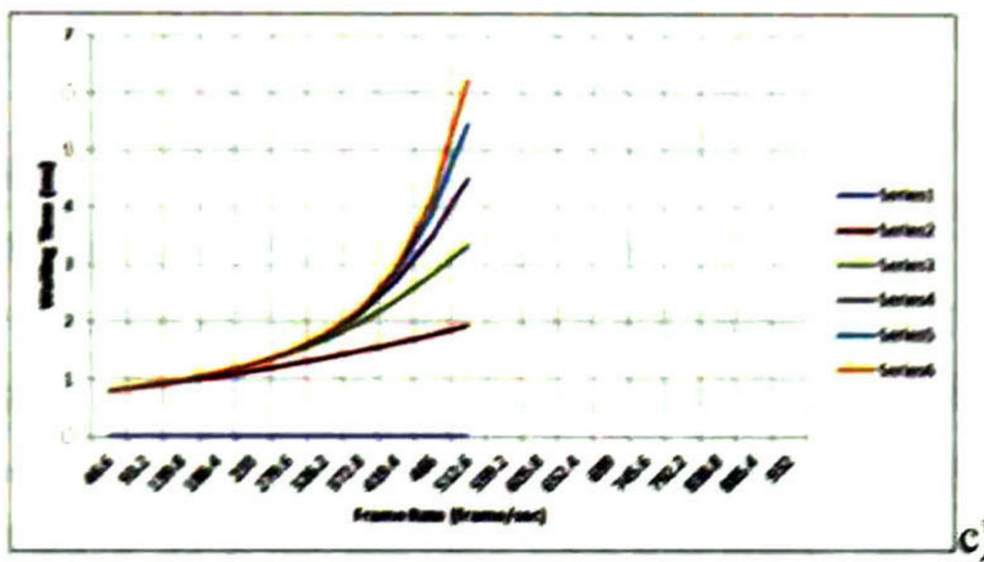
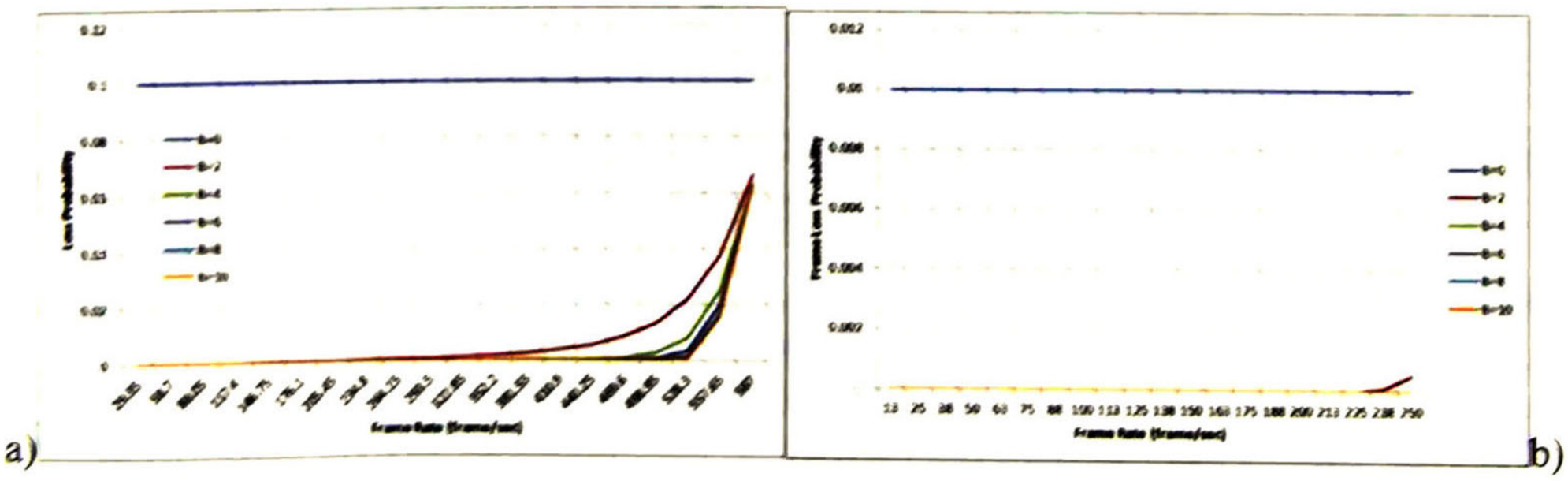
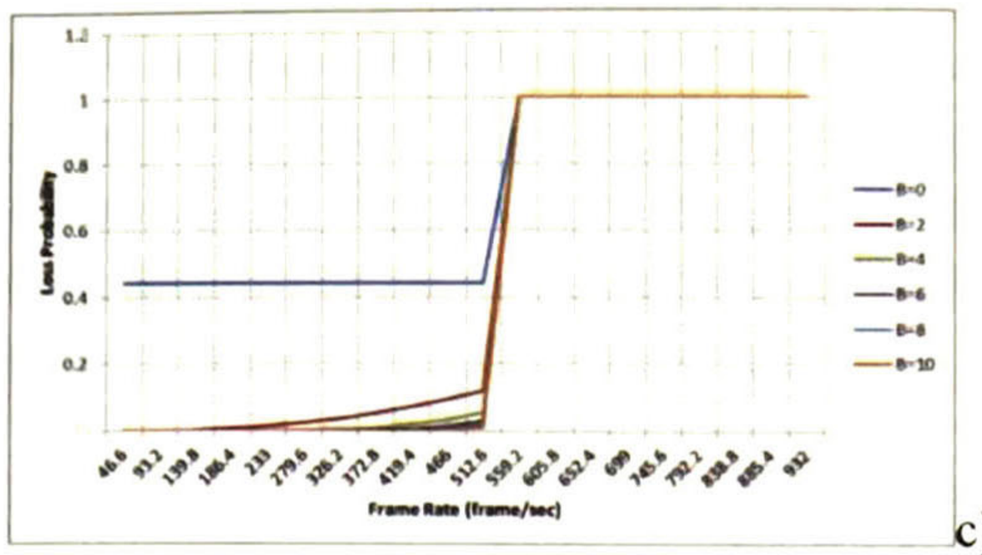


Fig. 33. Delay for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces

5.4.1.3 Frame Loss Probability analysis

Figure 34 shows the frame loss probability for the input buffers. For the 802.11 interface when buffer size employed is 0 the observed loss probability of 0.1, as shown in Figure 34a. For other buffer sizes the loss probability was reduced to almost 0 even at the maximum transmission frame rate of 500 frames/sec. In general terms, for the 802.15.4 interface the loss probability is not affected by the buffer size, as seen in Figure 34b. In the case of the CAN interface the observed loss probability reaches 1 when the implementation frame rate limit is employed. For lower frame rates and different buffer size values the CAN interface has a different behavior. Figure 34.c shows that when the 500 frames/sec is exceeded the loss probability is always 1. This means that the input interface cannot process such traffic amount and will drop the incoming frames.



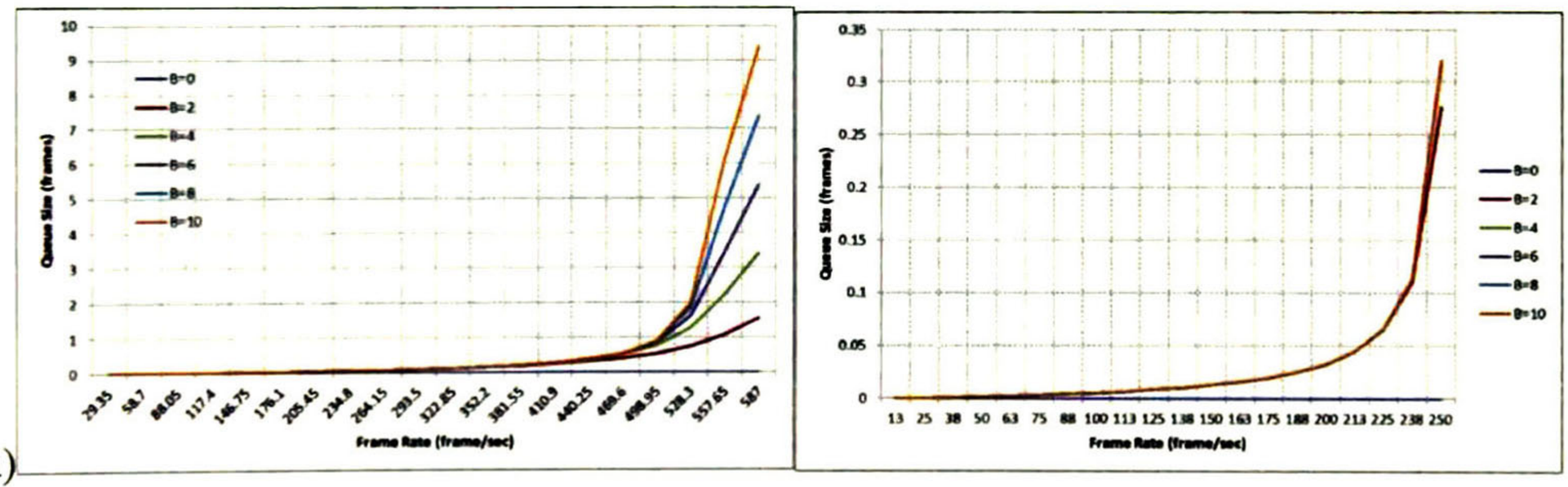


c)

Fig. 34. Loss probability for the M/M/1/B input buffer varying the frame rate and buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces

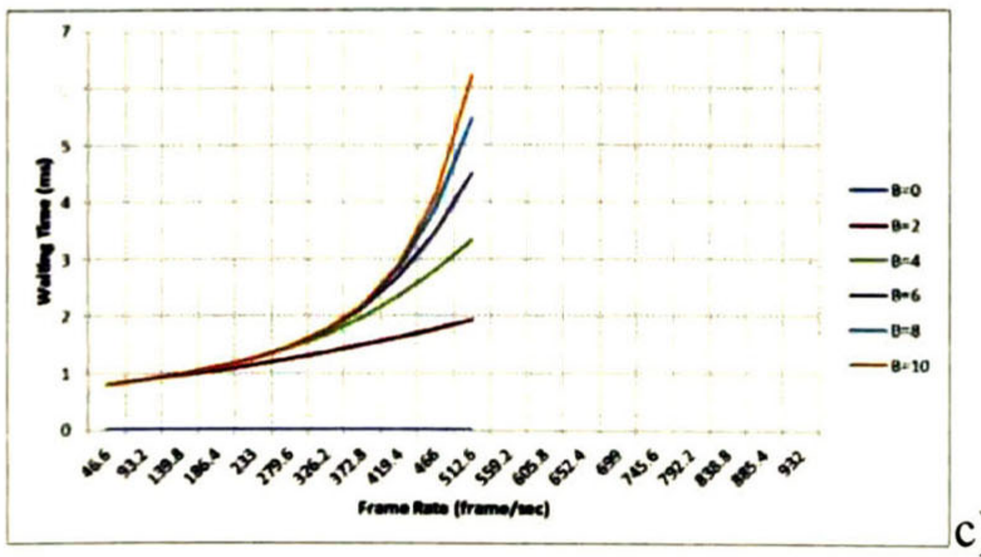
5.4.1.4 Queue size analysis

The queue size for input interfaces is shown in Figure 35. It is shown that for 802.11 interface the queue size increases only when the arrival frame rates are close to 500 frames/sec (see Figure 35.a). This queue size increment also affects the frame loss probability and the delay, but not significantly because the queue size never equals or exceeds the buffer size (the value of *B*). The same conditions are seen in the Figure 35.b for the 802.15.4 interface. In this case the increment in the queue size is minimal and therefore the frame loss probability and delay are affected at the minimum.



a)

b)



c)

Fig. 35. Buffer size for the M/M/1/B input buffer varying the frame rate buffer size for a) 802.11, b) 802.15.4 and c) CAN interfaces

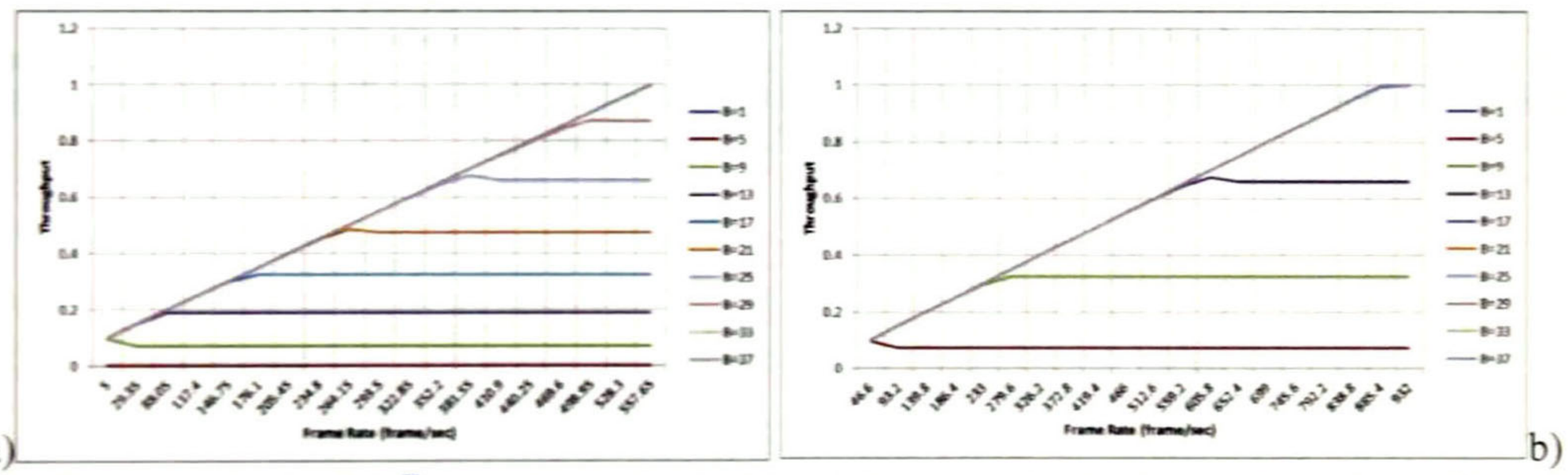
5.4.2 Output buffer analysis

For the output queue analysis different frame rates varying from 5 to 500 frames/sec were used. The upper limit is set to 500 frames/sec given that the input interfaces can only process this amount of frames. Different buffer sizes were used changing from 0 to 10. In sub-section 5.2.2.1 the performance of output interfaces using *frame aggregation* is analyzed, in sub-section 5.2.2.2 the output interfaces using *frame fragmentation* are studied and finally the output interfaces using *single frame encapsulation* are investigated in sub-section 5.2.2.3.

The frame aggregation is used in the 802.11 (aggregating 802.15.4 and CAN traffic) and 802.15.4 (aggregating only CAN frames) interfaces. For the former the aggregation rate for 802.15.4 traffic was set to 23 frames and 289 frames for the CAN traffic while the latter the aggregation rate was set to 12 CAN frames.

5.4.2.1 Throughput Analysis of Frame Aggregation

In Figure 36 is shown the throughput for the 802.11 interface aggregating frames of 802.15.4 (Figure 36.a) and CAN traffic (Figure 36.b). The same metric is shown in Figure 37 for the 802.15.4 interface aggregating CAN traffic. It is observable in all cases that if the buffer size is small, the throughput achieved is also small, even with the high frame rates. In contrast with the input buffers (see section 5.2.1.1), as the buffer can store more frames, the best the achieved throughput is.



a) b)

Fig. 36. Throughput for the M/M$^{[m]}$/1/B output buffer for 802.11 interface varying the frame rate and the buffer size (B) aggregating a) 802.15.4 and b) CAN traffic.

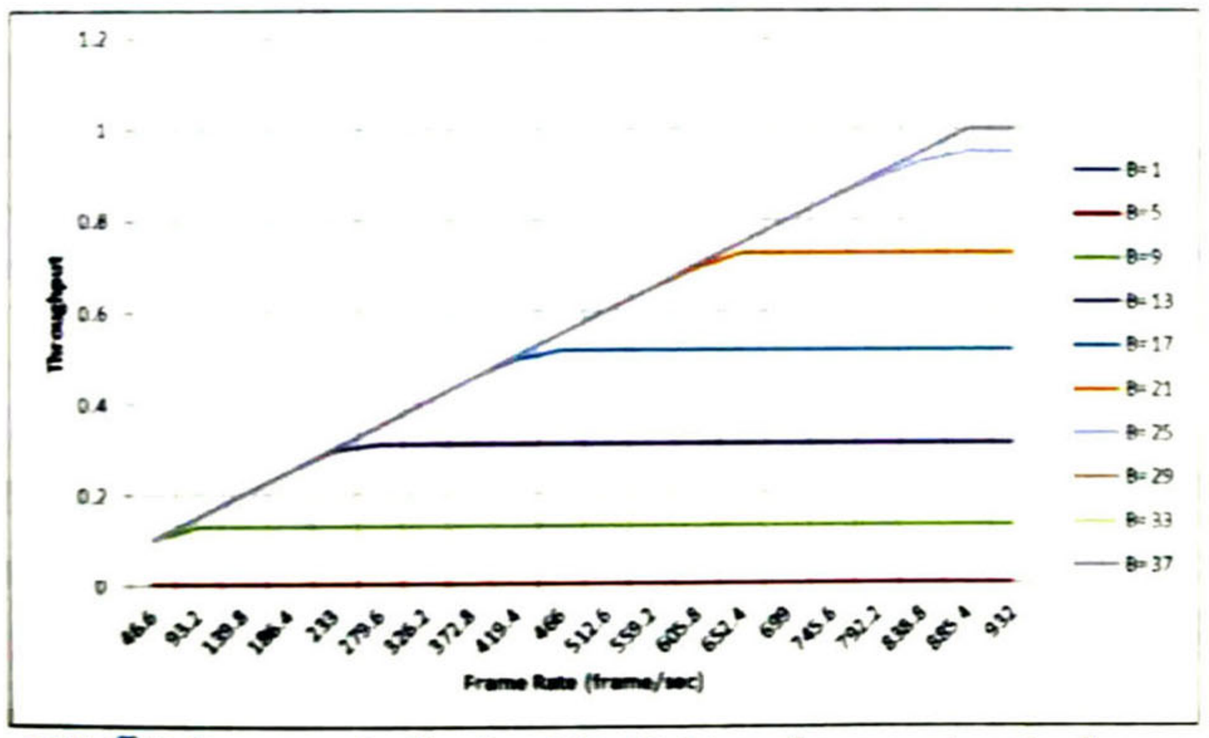


Fig. 37. Throughput for the M/M$^{[m]}$/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic

5.4.2.2 Delay Analysis of Frame Aggregation

The queuing delay is shown in Figures 38 (for 802.11 interface) and 39 (for 802.15.4 interface). In Figure 38 is observable that with all buffer size the delay starts in 10ms and in Figure 39 starts in 18ms. This is because the $M/M^m/1/B$ queuing system starts to serve the frames until the batch size (the *m* parameter) is reached. As the frame rate increases, the delay also increases and when the queue size is close to the buffer limit, the delay increases faster. In the Figures, the lines are truncated where the queue reaches its limit and it cannot serve any more frames. It is observable that using larger buffer sizes the delay remains low and the queue can attend higher frame rates.

The lower delay limit (observable in both Figures 38.a and 38.b in 10ms) is given by the size of forwarding frame and the output interface service time. This is clearly a disadvantage.

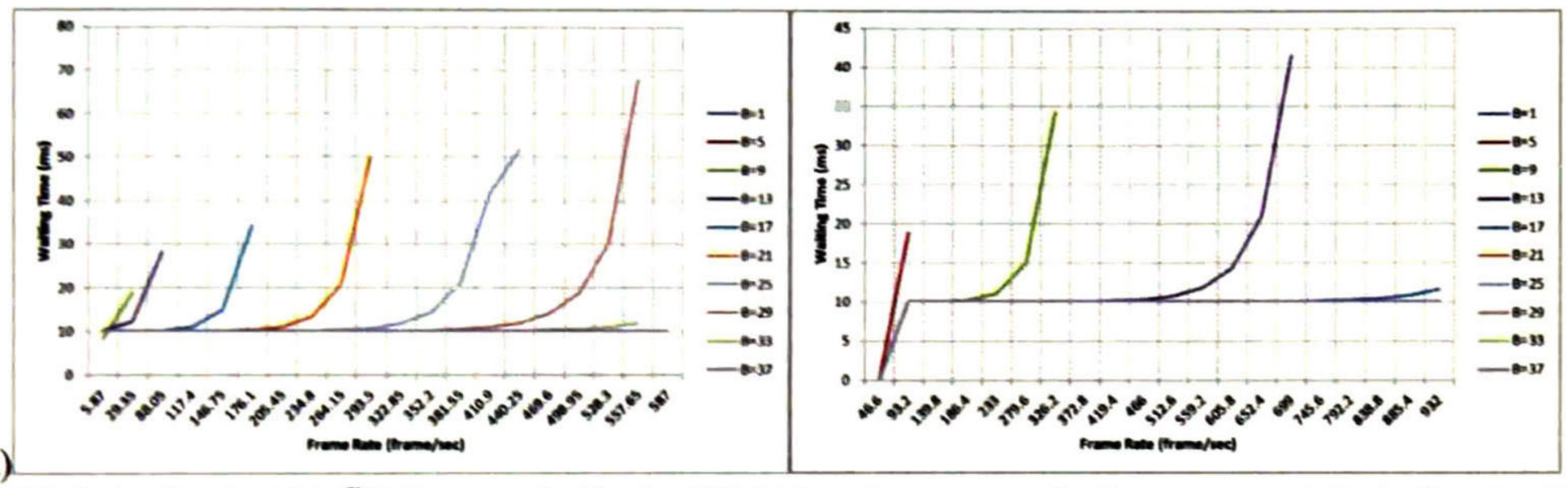


Fig. 38. Delay for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) aggregating a) 802.15.4 and b) CAN traffic.

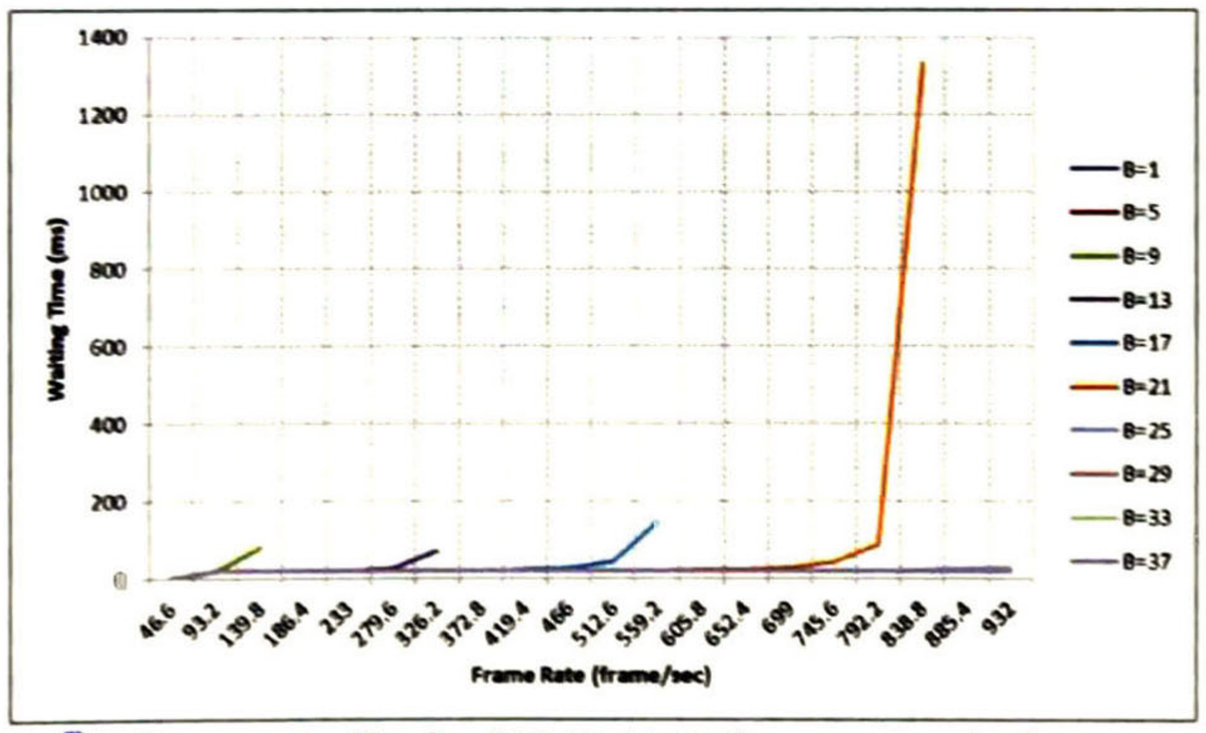
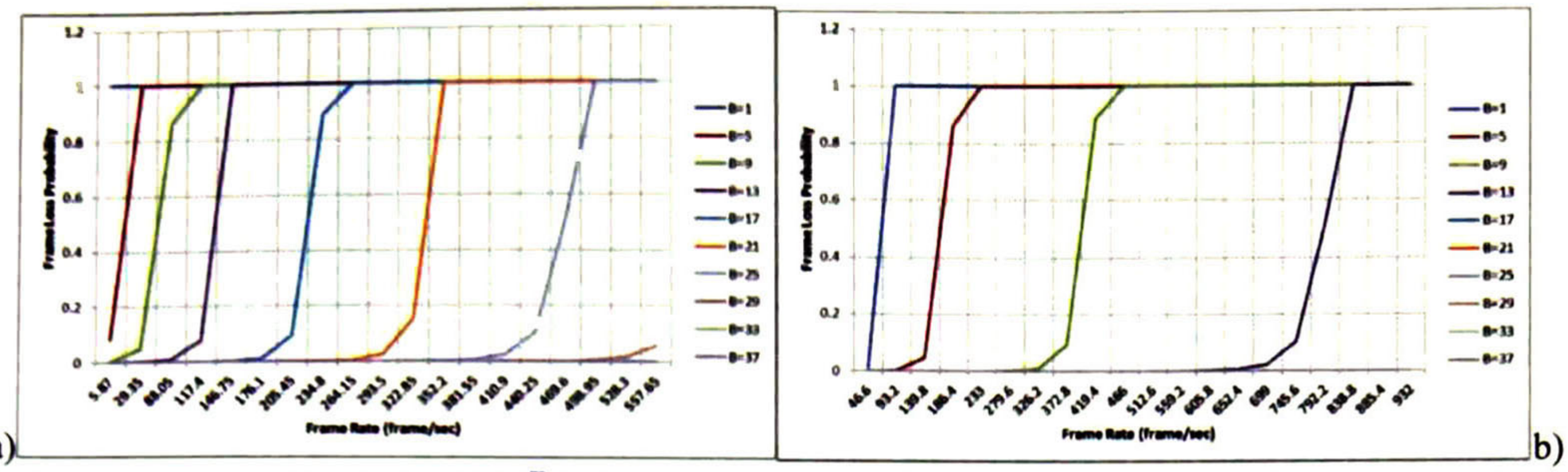


Fig. 39. Delay for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic

5.4.2.3 Frame loss probability Analysis of Frame Aggregation

In Figures 40 (the 802.11 interface) and 41 (the 802.15.4 interface) is observable that when the buffers are close to their limits, the frame loss probability increases dramatically and in the limit, the queue starts dropping frames. With greater buffer sizes, this increasing in the frame loss probability is slower. Given that the buffer size does not affect in negative way to the delay, using a larger buffer size can reduce the value of this two metrics, the delay and loss probability.



a) b)

Fig. 40. Frame loss probability for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) forwarding 802.15.4 traffic

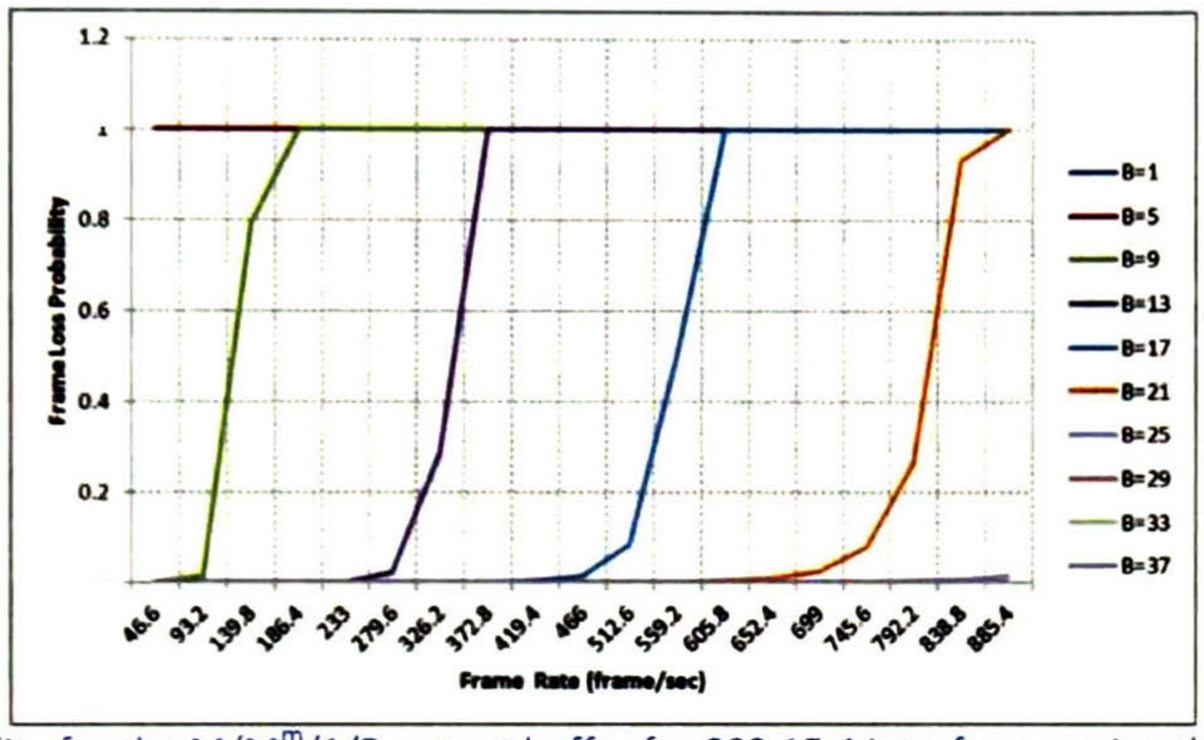
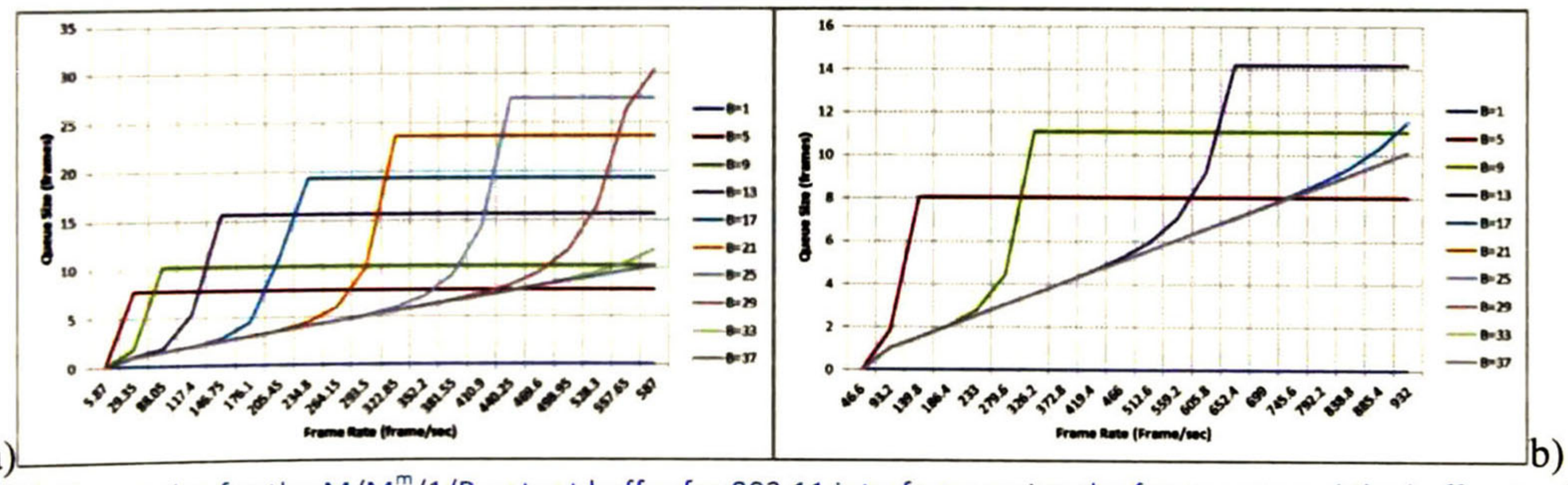


Fig. 41. Frame Loss Probability for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic

5.4.2.4 Frame loss probability analysis for Frame Aggregation

In Figures 42 is shown the queue size evolution for the 802.11 output interface and in Figure 43 the corresponding for 802.15.4 interface. It is observable that in 42.a and 42.b subfigures that when the buffer size is reached; it remains with the same value because the queue starts to drop the incoming frames. Also is observable that with larger buffer sizes, the queue size increases but reduces the loss probability and the delay.



a) b)

Fig. 42. Queue size for the $M/M^m/1/B$ output buffer for 802.11 interface varying the frame rate and the buffer size (B) forwarding 802.15.4 traffic

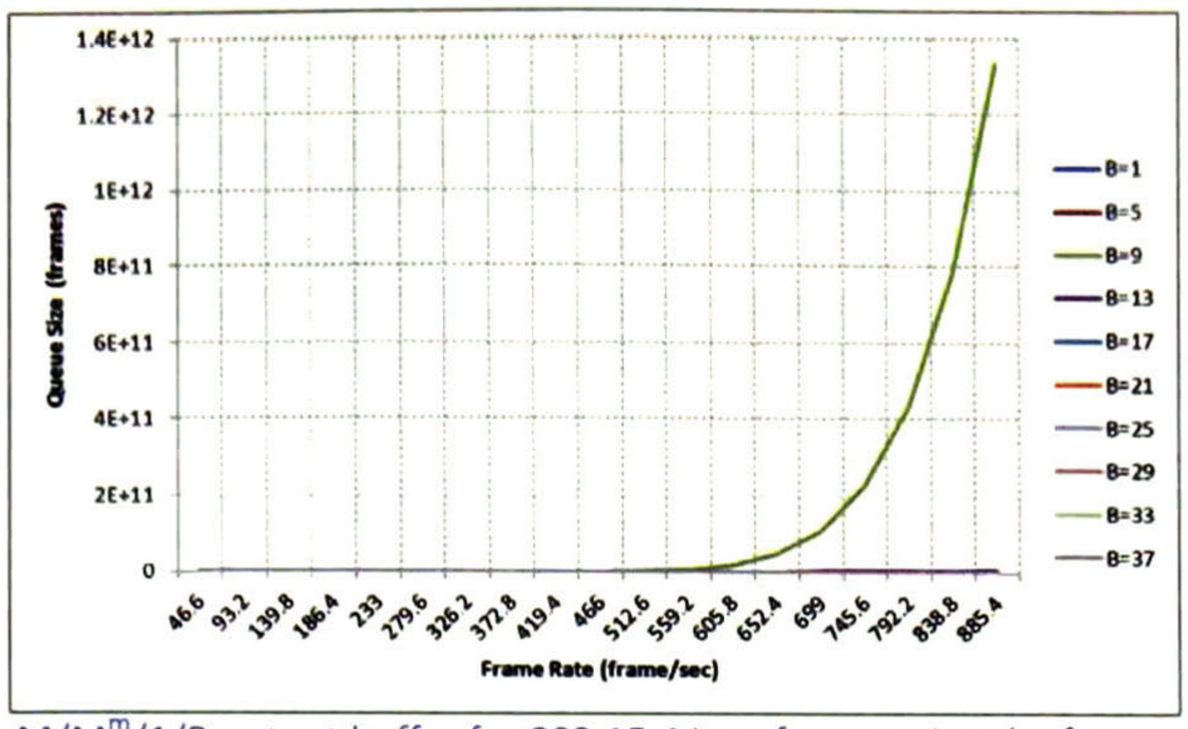


Fig. 43. Queue Size for the $M/M^m/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding CAN traffic

5.4.2.5 Performance Analysis of Frame Fragmentation

The frame fragmentation is used in the 802.15.4 (fragmenting 802.11 traffic) and CAN (fragmenting 802.11 and 802.15.4 frames) interfaces. For the first case, the 802.11 frames are segmented in 23 fragments each one. For the second case, the 802.11 frames are segmented in 289 and the 802.15.4 frames in 12 segments. The frame rate used was set from 25 to 500 frames

In Figure 44 the throughput for the frame fragmentation is shown. Given that a single 802.11 frame represents, for example 23 802.15.4 frames or 289 frames for CAN, the input traffic is multiplied. Then with the configurations mentioned above, the throughput is always the maximum for the interface.

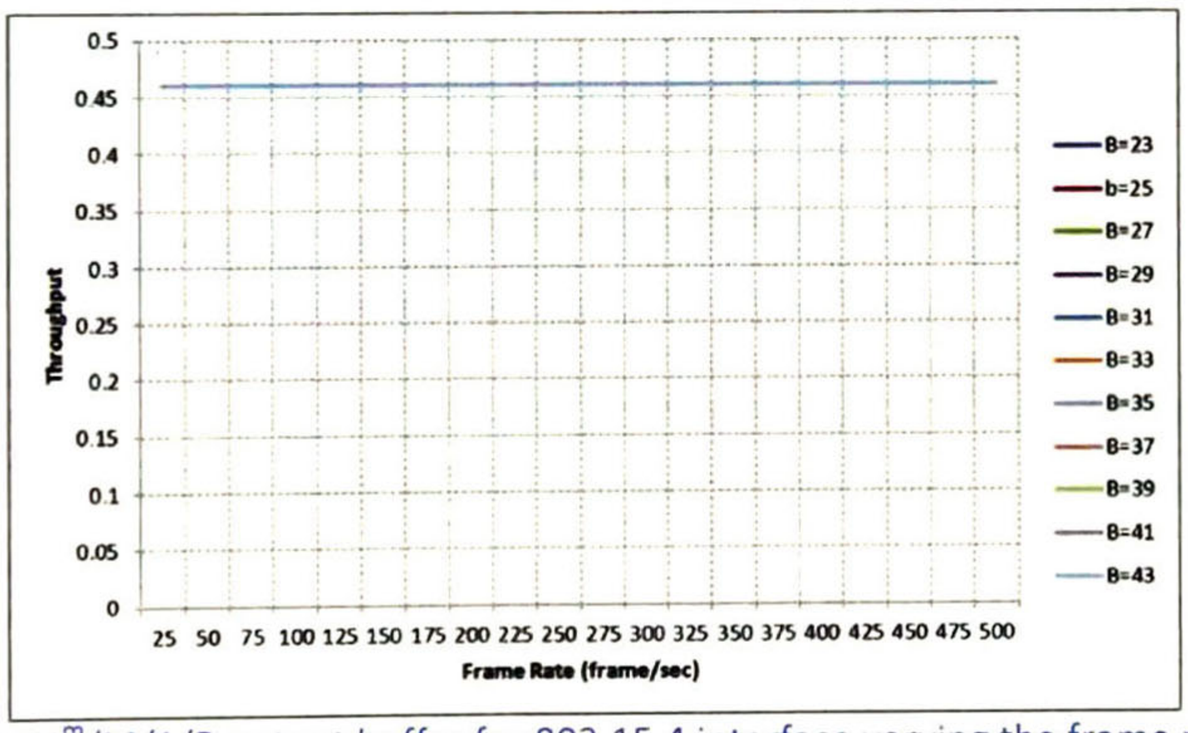


Fig. 44. Throughput for the $M^m/M/1/B$ output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic

The delay for fragmentation is shown in Figure 45. This is represented as flat lines because the arrival frame rate causes the buffer saturation and the time spent in the queue is always the maximum.

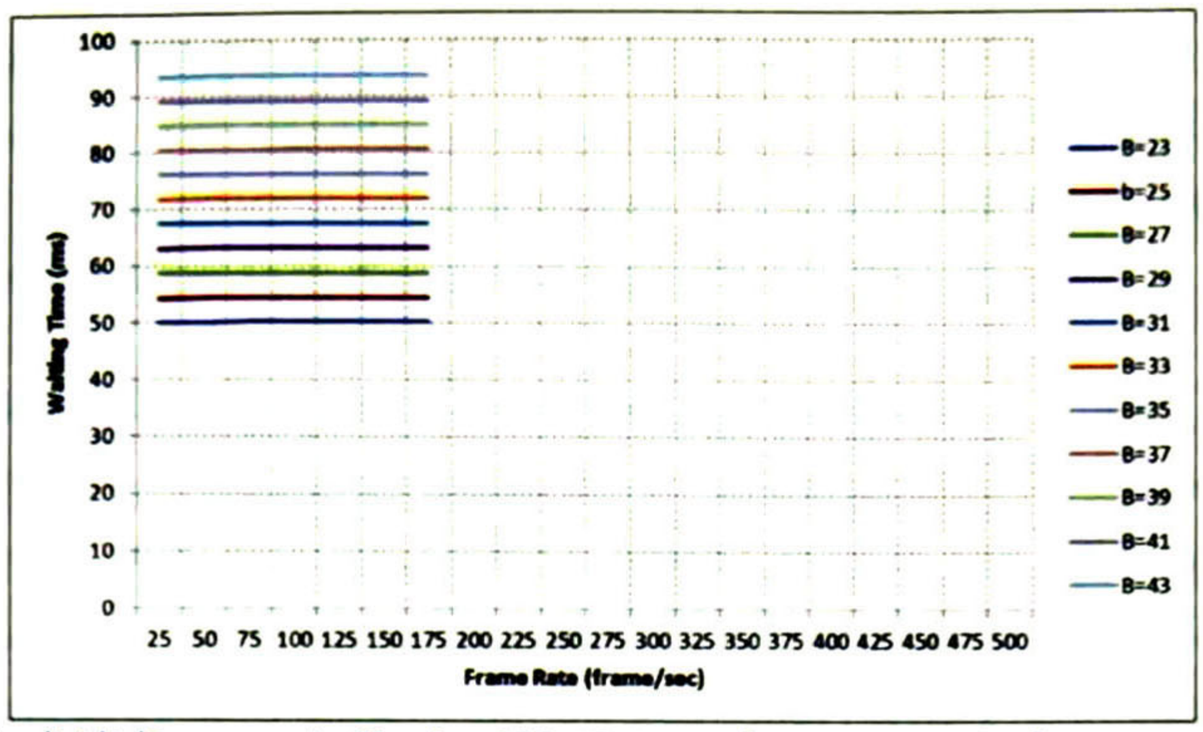


Fig. 45. Delay for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic

The Figure 46 shows the frame loss probability. For all combinations for this analysis, the probability to drop frames is 1. Again, this is because the arrival rates exceeding both the buffer size and the service capacity of the queue.

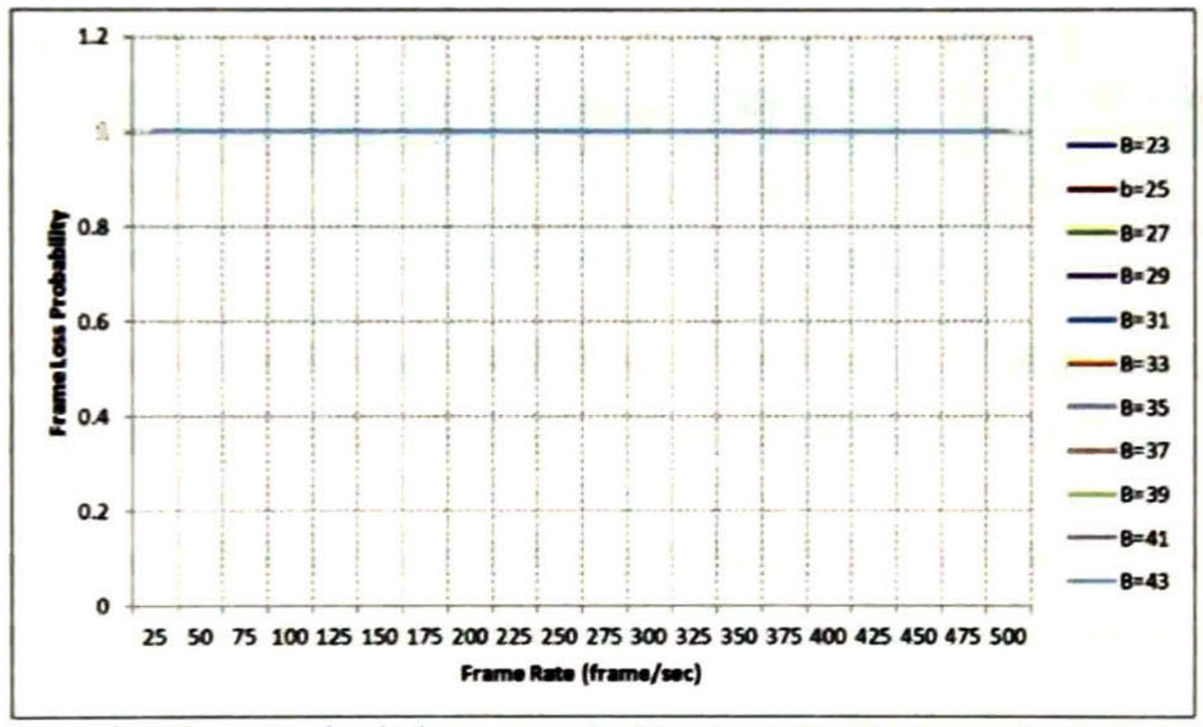


Fig. 46. Frame Loss Probability for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic

Similar results are expected for queuing system representing the CAN output interface, given that the arrival frame rate from 802.11 and 802.15.4 exceeds the queue capacity and service times are less than the 802.15.4 output interface.

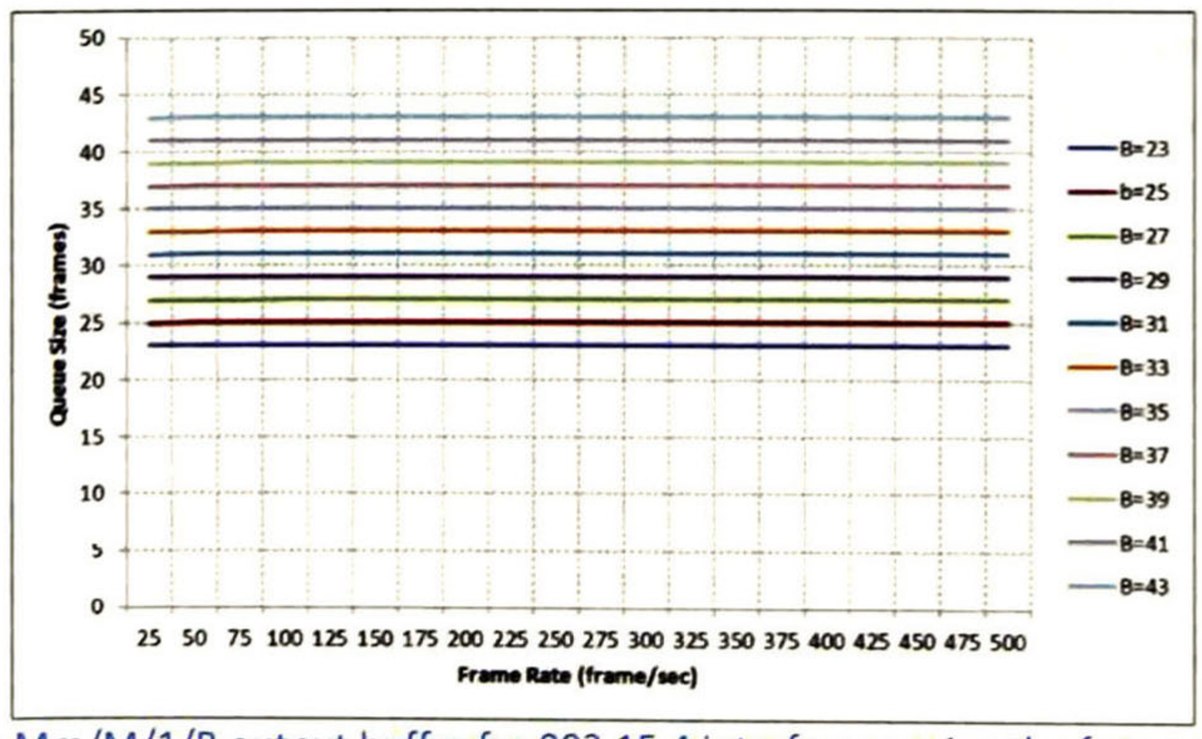


Fig. 47. Queue Size for the Mm/M/1/B output buffer for 802.15.4 interface varying the frame rate and the buffer size (B) forwarding 802.11 traffic

From this analysis is concluded that the buffer size affects in different way to the input and output queues. In the input queues, this parameter does not affect, even if the buffer size equals 1. In the other hand, the buffer size is determinant to reduce the delay and loss probability, if the queue is larger enough these two performance metrics can maintain a low value.

5.5 End to end transmission performance analysis

In this subsection the end to end delay and throughput are analyzed. Six different configurations are evaluated, varying the source and destination protocols and the encapsulation in the output interfaces in the bridge. The table 6 shows the used configurations. In this table SFE stands for Single Frame Encapsulation, FF for Frame Fragmentation and FA for Frame Aggregation.

Source Network	Destination Network	Output Encapsulation	Source Network	Destination Network	Output Encapsulation
CAN	802.11	SFE	802.11	CAN	SFE
CAN	802.11	FA	802.11	CAN	FF
CAN	802.15.4	SFE	802.15.4	802.11	SFE
CAN	802.15.4	FA	802.15.4	802.11	FA
802.11	802.15.4	SFE	802.15.4	CAN	SFE
802.11	802.15.4	FF	802.15.4	CAN	FF

Table 6. Transmissions configuration for end to end performance analysis

The test bed consists in five CAN nodes (as described in section 5.1), five 802.15.4 nodes, tow 802.11 nodes and a heterogeneous bridge implemented as described in section 5.3.

The set up for CAN consisted of 5 nodes (ECUs), emulated by the Vector© CANoe software in five stations connected to a physical bus of 40m of length. The payload size set to the maximum (8 bytes) value. The corresponding messages IDs for all nodes are the same on Table 5, but the periods where modified to vary the frame rate.

The 802.15.4 network consisted of 5 nodes using an Arduino© board with a Digi Xbee© pro series 1 module. In all experiments, the payload size was set to the optimum value. The frame rate was adjusted in all nodes to fit in the experiments requirement.

For 802.11 network two stations with wireless card and Linux operating system were employed. The payload size was set to the optimum value. As well in the CAN and 802.15.4, the frame rate was adjusted to fit in the experiment's requirements.

Finally, a heterogeneous bridge (as described in section 5.3) was employed to interconnect the three networks.

5.5.1 End to end delay analysis

Figure 48 shows the end to end delay for transmission originated in CAN nodes. Figure 48a shows this metric in the case where the 802.11 nodes are the destination. It is observable that increasing in delay values is minimum. Also, it is observable that using frame aggregation, the delay increases dramatically. This phenomenon is caused by the necessary waiting for frames to fulfill the 802.11

frame. This increasing is observable also in figure 48b, where the destination network uses 802.15.4 protocol. For this scenario, the delay suffers a higher increment given the lower frame rate of 802.15.4 interface and the waiting condition for the frame aggregation process.

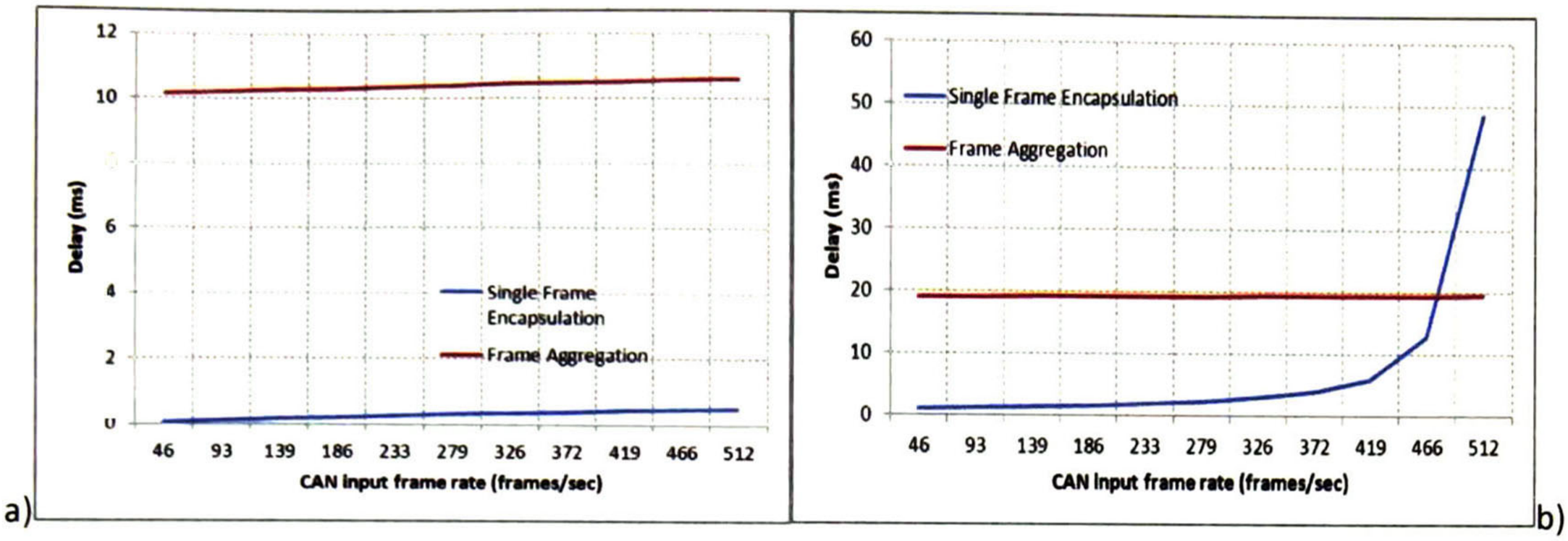


Fig. 48. End to end delay for transmissions from CAN to a) 802.11 and b) 802.15.4 nodes

Figure 49 shows the same metric where the transmissions are initiated in 802.11 nodes; figure 49a shows the delay with a 802.15.4 nodes as destination and it is observable that the delay is about 80 ms due the fragmentation process. The same is observable in figure 49b with CAN nodes as destination. The delay increases dramatically due the reduced payload in CAN frames and the fragmentation process.

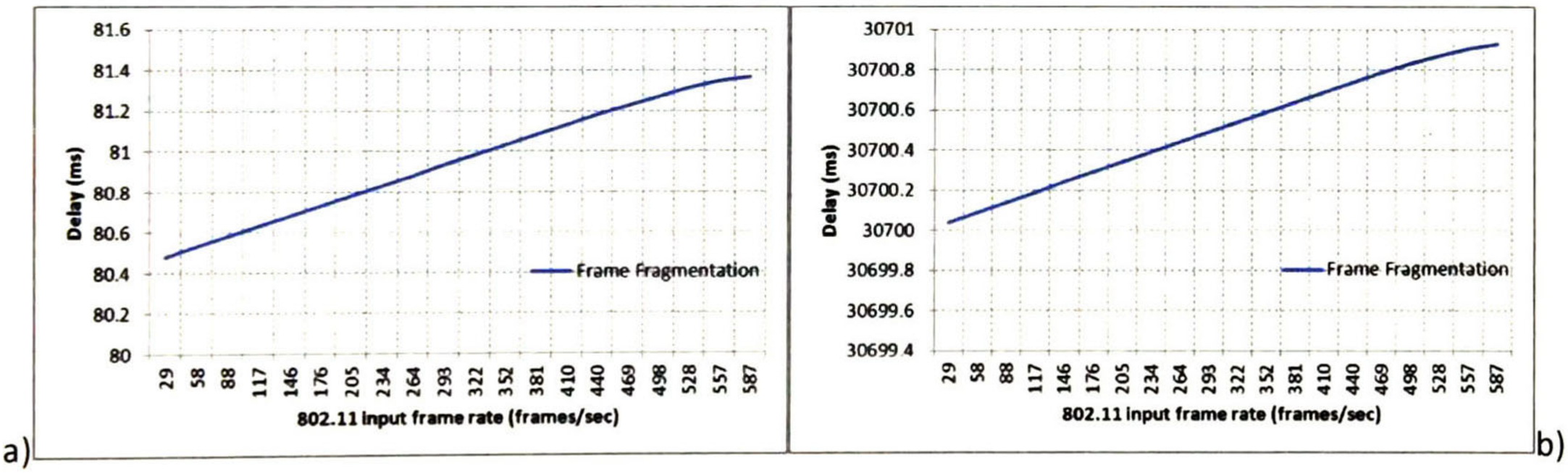


Fig. 49. End to end delay for transmissions from 802.11 to a) 802.15.4 and b) CAN nodes

Figure 50 shows the delay where the transmissions are initiated in 802.15.4 nodes; figure 49a shows the delay with a 802.11 nodes as destination and it is observable using single frame the delay has a minimum value and using frame aggregation the delay increases due the queuing process inherent to $M/M^m/1/B$ queuing system. Figure 49b shows delay with CAN nodes as destination. The delay increases dramatically due the reduced payload in CAN frames and the fragmentation process.

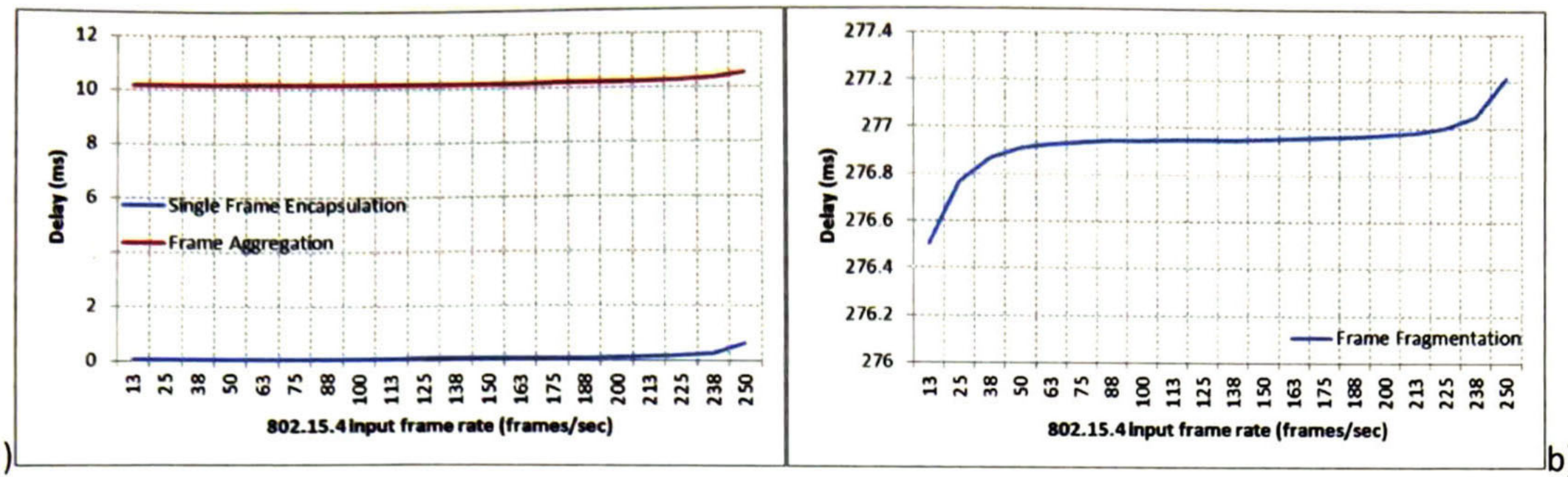


Fig. 50. End to end delay for transmissions from 802.15.4 to a) 802.11 and b) CAN nodes

5.5.2 End to end throughput analysis

Figure 51 shows the end to end throughput for transmission originated in CAN nodes. Figure 51a shows the throughput in the case where the 802.11 nodes are the destination and figure 52b the 802.15.4 are the destination nodes. It is observable that using frame aggregation or single frame encapsulation results in the same; due the frame rate of CAN nodes are insufficient to saturate the 802.11 or 802.15.4 data rates.

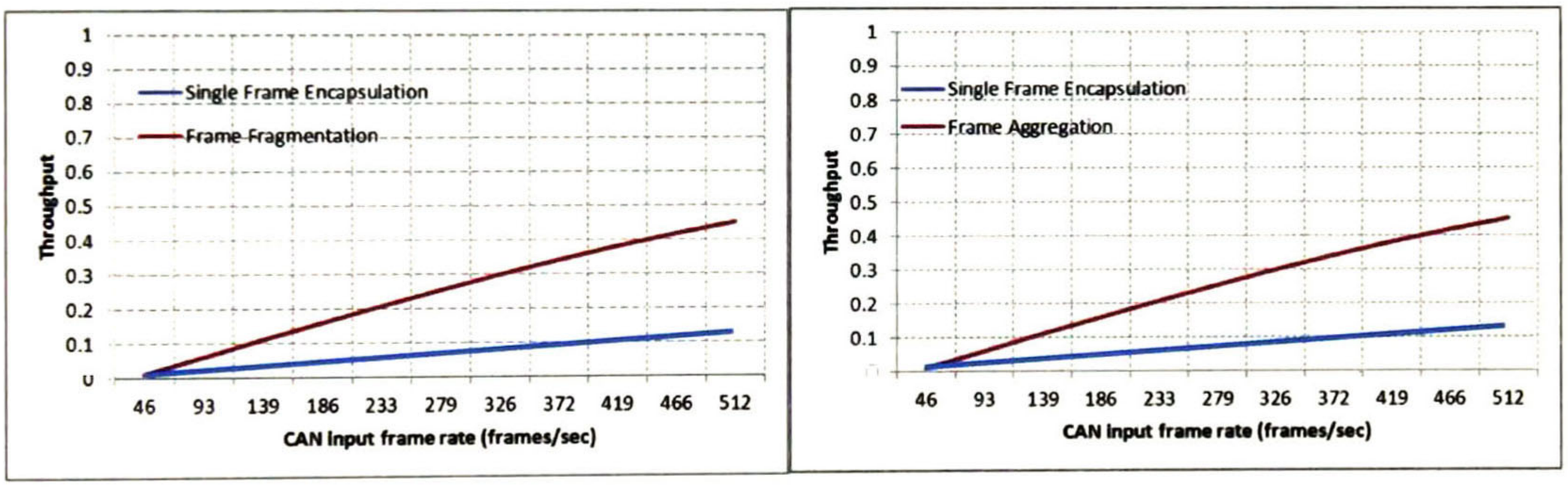


Fig. 51. End to end Throughput for transmissions from CAN to a) 802.11 and b) 802.15.4

Figure 52 shows throughput with transmissions originated in 802.11 networks and 802.15.4 (figure 52a) and CAN nodes (figure 52b) as destination nodes. It is observable that when the 802.11 traffic saturates the 802.15.4 frame capacity, reaches its maximum; same phenomenon is observable in CAN nodes, with the annotation that the saturation is reached sooner.

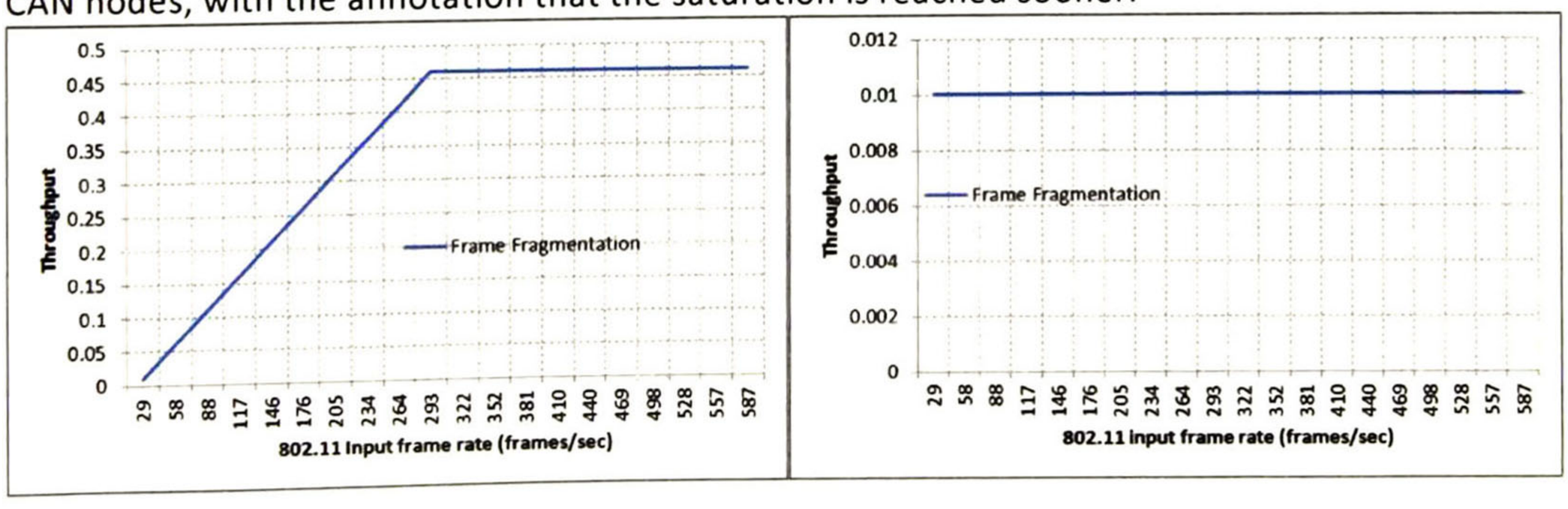


Fig. 52. End to end Throughput for transmissions from 802.11 to a) 802.15.4 and b) CAN

In figure 53a is observable the throughput for transmissions originated in 802.15.4 nodes and terminated in 802.11 nodes. This metric reaches its maximum using maximum frame rates in 802.15.4 nodes and frame aggregation. In figure 53b the CAN nodes are the destination, and the maximum throughput possible is reached using frame rates near to 50 802.15.4 frames per second.

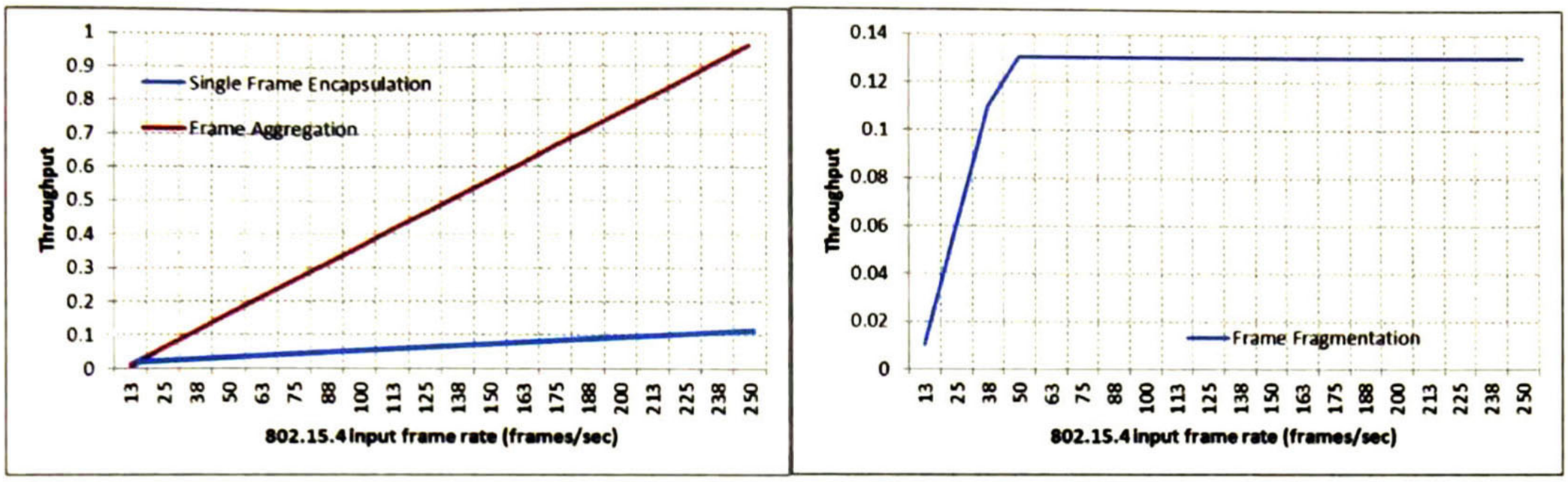


Fig. 53. End to end Throughput for transmissions from 802.15.4 to a) 802.11 and b) CAN

Chapter 6. Conclusions and Future Work.

6.1 Conclusions.

A model and analysis of a CAN node transmission was presented. Our proposed analytical model is based on the CLD approach which allows estimating the transmission delay, throughput and frame loss probability taking into account the BER, MAC and ARQ protocols, total frame size and encapsulation. The results estimated with this model were validated against a real network implementation and the comparative analysis shows the accuracy in our performance metrics estimations. Based on our estimations, is concluded that when using the maximum frame size the best throughput performance is obtained, with a higher delay and loss probability.

The proposed CAN model, unlike those reviewed in the literature, allows analyzing CAN transmissions not only in the worst case (a frame with the maximum length and amount of stuffing bits), but any other frame and transmission configuration.

Also a heterogeneous network mathematical model for the end to end network performance analysis was presented; composed by the bridge, CAN, 802.11 and 802.15.4 mathematical models, also based on the CLD approach. Besides, a design for the heterogeneous bridge was proposed. In section 4.1 the bridge's framing and queuing tasks (encapsulation, aggregation and disaggregation) and its behavior for heterogeneous environments have been formalized using process algebra. This formalization was used as the design specification for the design of the heterogeneous bridge which is presented describing the full functionality of each component of the bridge.

The analysis for the heterogeneous network was divided in 3 parts: i) the input and output queuing analysis at the bridge, where was found that in the input queues the buffer size do not affect the performance metrics, hence is possible to use the minimum buffer size without increasing the delay or reducing the throughput. In the other hand, the output queues are highly dependent on the buffer size. The buffer size must be at least equal to aggregation rate or fragmentation rate, but if the buffer size is higher, the performance metrics are positively affected. ii) In the entire bridge analysis, was found that the throughput in the bridge is limited by the lowest capacity network interface. For instance, the CAN traffic did not saturate the bridge or the 802.11 network even with the highest CAN frame rate. From this, it is observable that the end to end throughput is limited by the CAN network. In the opposite direction, is possible to see that the delay dramatically increases in the bridge (and consequently in the end to end delay) because the 802.11 traffic load is higher than CAN channel capacity. Under the same network conditions the end to end throughput is decreased.

The acceptable level of error in the numerical results showed the satisfactory precision of the proposed mathematical framework on modeling heterogeneous network environments composed by the mentioned network domains.

It was found that implementing the aggregation process with a strict $M^m/M/1/B$ queue can be a problem given that these systems wait until *m* frames arrives to complete the forwarding frame and increases the delay in the output queues.

6.2 Future Work

Teletraffic modeling.

- To model the frame aggregation and frame fragmentation with a more flexible queuing system than the used in this research in order to reduce the waiting time while the queue composes the forwarding frame.
- Include other protocols such X10, MOST or Zwave to expand the bridge possibilities for sensor networks. In this sense, to model other networks such 802.16 or 802.11p to include MAN networks in the heterogeneous bridge.

Bridge Design

- Including security modules in the bridge will enhance its functionality, the commercialization possibility and applications.
- To include a bridge communication protocol to expand the interconnection possibilities using multiple heterogeneous bridges in a network.

Implementation

- Finish the initial design implementation, including the adaptability modules and the changes to the mathematical modeling mentioned above.

References

[1] Weiser, M. "The computer for the 21st Century" Sci. Amer. Vol. 3 Sept. 1991

[2] P. Marti, H. Hautop. "Ambient Intelligence Solutions for Edutainment Environments" Proceedings of AI*IA 2003 (Eight National Congress of Associazione Italiana per l'Intelligenza Artificiale). Springer-Verlag. 2003

[3] K.C. Lee, H.H. Lee. "Network-based Fire-Detection System via Controller Area Network for Smart Home Automation" IEEE Transactions on Consumer Electronics Volume 50, 2004

[4] K. Grill, S. Yang, F. Yao and X. Lu. "A ZigBee-Based Home Automation System" IEEE Transactions on Consumer Electronics Volume 55, 2009

[5] A. Rakotonirainy and R. Tay. "In-Vehicle Ambient Intelligent Transport Systems (I-VAITS): Towards an Integrated Research". Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems. 2004.

[6] P. Papadimitratos, A. Mishra, "A cross-layer design approach to enhance 802.15.4" , IEEE MILCOM 2005, Atlantic City, NJ, USA, October 17-20, 2005

[7] T. Canli, N. Abdesselam, A. Khokhar, "A cross-layer optimization approach for efficient data gathering in wireless sensor networks", Networking and Communications Conference, 2008. INCC 2008. IEEE International

[8] M. Vuran, I. Akyldiz, "Cross-layer packet size optimization for wireless terrestrial, underwater and undergroung sensor networks", INFOCOM 2008. The 27th Conference on Computer Communications. IEEE

[9] M. Vuran, I. Akyldiz, "Cross-layer analysis of error control in wireless sensor networks" SECON 2006 proceedings.

[10] V. Srivastava and M. Montani, "Cross Layer Design, a Survey and the Road Ahead", Communications Magazine, IEEE, 2005

[11] J. Mehlman, "Cross-Layer Design: A Case for Standardization", Research Report, Electrical Engineering Department, Stanford University, 2008

[12] J. Roese, "Switched LANs, Implementation, Operation, Maintenance", Ed. McGraw Hill, Computer Communications. 1998

[13] IEEE, "802.1D® IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture", 2002

[14] R. Seifert, "The Switch Book", Ed. Wiley Publishing Inc. 2000

[15] F. Gebali, "Analysis of computer and communication networks", Ed. Springer. 2008

[16] T. Lammle, "CCNA, Cisco Certified Network Associate Study Guide", Fifth edition, Ed. Wiley Publishing Inc. 2005

[17] IEEE. "IEEE Standard 802.3™-2005"

[18] IEEE. "IEEE Standard 802.11™-2007"

[19] IEEE "IEEE Standard 802.15.4™-2006"

[20] R. Bosch, "CAN Specification Version 2.0", Robert Bosch GmbH, 1991

[21] C. Bayilmis, I. Erturk, C. Ceken, I Ozcelik. "A CAN/IEEE 802.11b wireless LAN local bridge design" Computer Standards & Interfaces, Elsevier 2007

[22] I. Erturk "A new method for transferring CAN messages using wireless ATM" Journal of Network and Computer Applications, Elsevier, 2005

[23] I Ozcelik, "Interconnection of CAN segments through IEEE 802.16 wireless MAN", Journal of Network and Computer Applications, Elsevier, 2008

[24] I. Ozcelik, H. Ekiz "Design, implementation and performance analysis of the CAN/ATM local bridge" Journal of Computer Standards & Interfaces Elsevier 2007

[25] J. Leal, A. Cunha, M. Alves, A. Koubâa, "On a 802.15.4/Zigbee to 802.11 Gateway for the ART-WiSe Architecture" ETFA Conference on Emerging Technologies and Factory Automation, 2007

[26] H. Ekiz, A. Kutlu, M.D. Baba, E.T. Powner, "Performance analysis of a CAN/CAN bridge" IEEE, Fourth International Conference on Network Protocols (ICNP'96), 1996

[27] A. Dunkels, J. Alonso, T. Voight "Making TCP/IP viable for wireless sensor networks", Proceedings of the first European Workshop on Wireless Sensor Networks , January 2004

[28] A. Dunkels, J. Alonso, T. Voight, "Connecting wireless sensor networks with TCP/IP networks", proceeding of the Second International Conference on Wired/Wireless internet communications February 2004

[29] M. Ditze, R. Bernahardi, G. Kamper, P. Aktenbernd "Porting the Internet Protocol to the Controller Area Network", Workshop in Real Time LANs in the Internet Age, 2003

[30] R. Intanagonwiwat, Govindan, R. Estrind D. "Directed Difusion: A Scalable and Robust Communication Paradigm for Sensor Networks" proceedings ACM MobiCom, 2000

[31] M. Johanson, L. Karlson, T. Risch. "Relaying Controller Area Network Frames over Wireless Internetworks for Automotive Testing Applications". Fourth International Conference on Systems and Network Communications. IEEE Computer Society, 2009

[32] L.A. Castro de Almeida and C. A. dos Reis Filho. "Latency evaluation in Bluetooth-CAN Dual Media Sensor Network" IEEE International Conference on Industrial Technology (ICIT), 2010

[33] M. Mirabella, "A Hybrid Wired/Wireless Networking Infrastructure for Greenhouse Management". IEEE Transactions on Instrumentation and Measurement, 2011

[34] Silva H. Figueiredo, L. Robaldão C. Pereira, A. "Wireless Network, Wifi/Wimax Handover" 4th International Conference on Systems and Networks Communications, IEEE, 2009

[35] El- Azouzi r., Sabir E., Samanta S. El. Khoury R. Bouyakhf E. "An end to endQoS framework for IEEE 802.16 and Ad-Hoc Integrated Networks" ACM, 2009

[36] R. Fantacci, D. Tarchi "Bridging Solutions for a Heterogeneous WiMAX-WiFi Scenario", Journal of Communications and Networks, 2006

[37] S. Figuereido., I. Robaldao., C. Pereira. "Wireless network interoperability, Wifi/WiMax Handover", 4th International Conference on Systems and Network Communications, IEEE, 2009

[38] X. Yang, J. Liu, F. Zhao and N.H. Vaidya, "A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning," Mobile and Ubiquitous Systems: Networking and Services, pages 114-123, August 2004.

[39] R. Aquino-Santos, V. Rangel-Licea, A. Edwards, "Inter-Vehicular Communications Using Wireless Ad-Hoc Networks." Ingeniería, investigación y tecnología, 9(4), 319-328., 2008.

[40] R. Xia, Ch. Yen, D. Zhang. "Vehicle to Vehicle and Roadside Sensor communication for Intelligent Navigation", 6th International Conference on Wireless Communications Networking and Mobile Computing, 2010

[41] R. Davis, S. Kollman, V. Polex, F., Slomka, "Controller Area Network (CAN) schedulability analysis with FIFO Queues" 23rd Euromicro Conference on Real-Time Systems, 2011

[42] T. Nolte, H. Hansson, C. Norström, "Probabilistic Worst Case Response- Time Analysis for Controller Area Network" Proceedings of the 9th IEEE Real Time and Embedded Technology and Applications Symposium. 2003

[43] H. Zeng, M. Di Natale, P. Giusto, A. Sanigiovanny-Vincentelli, "Statisticall Analysis of Controller Area Network Message Response Times", IEEE International Symposium on Industrial Embedded Systems, 2009.

[44] M. Di Natale, H. Zeng, "System Identification and Extraction of Timing Properties from Controller Area Network (CAN) Message Traces", IEEE Conference on Emerging Technologies and Factory Automation (ETFA), 2010

[45] G. Bianchi. "Performance analysis of the IEEE 802.11 Distributed Coordination Function", IEEE Journal on Selected Areas in Communications, March 2000

[46] D. Malone, K. Duffy, D. Leith.Modeling the 802.11 Distributed Coordination Function in nonsaturated heterogeneous traffic contitions, IEEE COMMUNICATIONS LETTERS, AUGUST 2005

[47] K. Duffy, D. Malone, D. Leith. "Modeling the 802.11 Distributed Coordination Function in nonsaturated heterogeneous traffic contitions." IEEE/ACM Transactions on Networking, Feb. 2007

[48] F. Lin, W. Liu, "Frame-aggregated link adaptation protocol for next generation wireless local area networks" EURASIP Journal on Wireless Communications and Networking, Hindawi Publishing Corporation, 2010, Article ID 164651

[49] J. Mišić, S. Shafi, V. Mišić. "The impact of MAC parameters on the performance of 802.15.4 PAN", Ad Hoc Networks, Elsevier, 2004

[50] C. Y. Jung, H. Y. Hwang. D. K. Sung, G. U. Hwang. "Enhanced Markov chain model and throughput analysis of the slotted CSMA/CA for IEEE 802.15.4 under unsaturated traffic conditions" IEEE transactions on Vehicular Technology, Jan. 2009

[51] T.O. Kim, J.S. Park, H. J. Chong, K. J. Kim, B. D. Choi. "Performance analysis of IEEE 802.15.4 Non-beacon mode with the unslotted CSMA/CA" IEEE Communications Letters, April 2008.

[52] P. Papadimitratos, A. Mishra, D. Rosenburgh. "A Cross-Layer Design approach to enhance 802.15.4" IEEE Military Communications Conference, 2005.

[53] T.O. Kim, J. S. Park, B. D. Choi. "Analytic model of IEEE 802.15.4 with download traffic", 21st International Conference on Advanced Information Networking and Applications Workshops, 2007

[54] IEEE, "802.1D IEEE Standard for Local and metropolitan area networks Media Access Control (MAC) Bridges", 2004

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.
UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Modelado y desarrollo de multipasarelas para redes heterogeneas

del (la) C.

Abraham Jair LÓPEZ VILLALVAZO

el día 04 de Abril de 2014.

Dr. Félix Francisco Ramos Corchado
Investigador CINVESTAV 3B
CINVESTAV Unidad Guadalajara

Dr. Ramón Parra Michel
Investigador CINVESTAV 3B
CINVESTAV Unidad Guadalajara

Dr. Mario Angel Siller González
Pico
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara

Dra. Susana Ortega Cisneros
Investigador CINVESTAV 2C
CINVESTAV Unidad Guadalajara

Dr. Héctor Alejandro Durán Limón
Profesor-Investigador
Universidad de Guadalajara