University of Porto
Faculdade de Engenharia
FEUP

Supporting Real-Time Communication in CSMA-Based Networks: The VTP-CSMA Virtual Token Passing Approach

by

Ricardo Alexandre Reinaldo de Moraes

A dissertation submitted in partial fulfilment of the requirements for the degree of Doctor in Electrical and Computer Engineering

Supervisor: Francisco Vasques
Co-supervisor: Paulo Portugal

March 2007

Abstract

There is a current trend towards the use of CSMA-based networks to support real-time (RT) communication in industrial environments. In the last few years Ethernet, for example, has emerged as a *de facto* communication standard for industrial communication. In addition, there will be a major increase in the demand for high performance industrial wireless networking in the next few years. This tendency comes in the wake of an increasing use of wireless communication both in office and domestic environments and in public hot-spots. Therefore, it is reasonable to expect that in the near future, the widespread availability of wireless solutions will generate a similar *de facto* standard for industrial wireless communications. The IEEE 802.11 family of protocols will definitely be one of the main contenders for industrial wireless solutions.

Communication requirements in industrial environments are very specific. In addition to multipurpose traffic similar to the traffic found in office/home environments, there is the requirement to support real-time traffic. This real-time traffic is typically associated with control applications, where real-time control data must be periodically transferred between sensors, controllers and actuators according to strict transfer deadlines. Emerging VoIP applications will also stress the real-time usage of industrial communication systems. Therefore, one of the fundamental questions that must be addressed when setting up a real-time communication system is: "How to guarantee the timing requirements of RT control data, when the communication medium is shared with timing unconstrained traffic?"

The main objective of this thesis is to address this question. That is, to propose mechanisms that enable the support of RT communication services in CSMA-based communication infrastructures, specifically when such infrastructures are also being used to support multipurpose data transfer applications.

The target is to address the timing requirements of typical industrial applications.

The following assumptions are made in this thesis: (i) a communication environment with timing unconstrained stations (ST stations) and real-time stations (RT stations) sharing the same communication medium (wired or wireless); (ii) the network load imposed by the set of ST stations is out of the sphere-of-control of the RT communication architecture. Additionally, it is also considered that no hardware/software changes should be necessary to ST stations, due to the large installed base of standard IEEE 802.3/802.11 devices.

Thus, a new RT-communication approach (VTP-CSMA approach) has been proposed, which is based on the use of traffic separation mechanisms. Such mechanisms are able to prioritize RT-traffic over multipurpose traffic, without directly controlling the latter. That is, instead of controlling *all* the traffic generated by *all* the stations, the proposed VTP-CSMA approach controls only the traffic generated by the RT stations. The VTP-CSMA approach forces the collision resolution in favor of the RT stations. Thus, it enables the fulfillment of the RT communication requirements.

Resumo

Actualmente, existe uma forte tendência para o uso de redes baseadas no protocolo CSMA para suportar comunicações de tempo-real (TR). Nos últimos anos, as redes Ethernet tornaram-se um padrão *de facto* para ambientes industriais. Além disso, a procura de redes sem fio de alto desempenho crescerá de uma forma significativa durante os próximos anos. Esta tendência é uma consequência da crescente utilização de comunicações sem fio em ambientes de escritório, pontos públicos de acesso e domésticos. Portanto, é provável que num futuro próximo, a ampla disponibilidade de soluções de redes sem fios irá também gerar um padrão *de facto* para comunicação sem fios em ambientes industriais, onde o conjunto de protocolos normalizados IEEE 802.11 será um dos principais candidatos.

No entanto, os requisitos de comunicação para ambientes industriais são muito específicos. Para além de tráfego genérico (por exemplo, tráfego de dados e multimédia) similar aquele encontrado em ambientes de escritório/doméstico, existe tráfego com requisitos de tempo-real. Este tráfego está tipicamente associado a aplicações de controlo, para as quais os dados de controlo devem ser periodicamente transferidos entre sensores, controladores e actuadores de acordo com metas temporais de tempo-real. Em consequência, uma das questões fundamentais que devem ser abordadas, especialmente quando se utilizarem redes sem fio em ambientes industriais é: "Como garantir que os requisitos temporais dos dados de controlo são respeitados, quando o meio de comunicação é partilhado com tráfego genérico não controlado?"

O principal objectivo desta tese é responder a esta questão. Ou seja, desenvolver novos mecanismos para suportar serviços de comunicação de tempo-real sobre uma infra-estrutura de comunicações baseada no protocolo CSMA, a qual também é usada para suportar tráfego multimédia e para a transferência de da-

dos em *background*. A meta desta tese é endereçar aplicações com requisitos de tempo-real típicos.

Os seguintes pressupostos são considerados nesta tese: (i) ambientes de TR constituídos por estações normalizadas (estações ST) e estações de tempo-real (estações TR), que partilham o mesmo domínio de comunicação (cablados ou sem fios); (ii) a carga imposta na rede pelo conjunto de estações ST está fora da esfera-de-controlo da arquitectura de comunicação de TR. Adicionalmente, considera-se que nenhuma alteração de hardware/software deve ser necessária nas estações ST, devido à grande quantidade instalada de dispositivos IEEE 802.3/802.11 normalizados.

Em consequência, no âmbito desta tese uma nova abordagem de comunicação de TR (abordagem VTP-CSMA) é proposta, a qual é baseada em mecanismos de separação de tráfego. Estes mecanismos são capazes de prioritizar o tráfego TR sobre os outros tráfegos, sem ter a necessidade de controlar directamente o tráfego não TR. Ou seja, em vez de controlar *todo* o tráfego gerado por *todas* as estações, a abordagem proposta controla unicamente o tráfego gerado pelos dispositivos de TR, de tal forma que a resolução de colisões é forçada em favor das estações TR, cumprindo os requisitos de comunicação de TR.

Résumé

Actuellement, il existe une forte tendance vers l'utilisation des réseaux basés sur le protocole CSMA pour le support de communications temps-réel (TR). Dans les dernières années, les réseaux Ethernet sont devenus une norme *de facto* pour les environnements industriels. Par l'autre coté, l'exigence de haute performance par rapport aux réseaux sans fil grandira d'une forme significative pendant les prochaines années. Cette tendance est une conséquence de la croissante utilisation de communications sans fil dans la bureautique, les points publics d'accès et les environnements domestiques. Donc, c'est bien probable que dans un avenir proche, la disponibilité des solutions sans fils produit aussi une norme *de facto* pour la communication sans fil dans les environnements industriels. Dans ce cas, la famille de protocoles normalisés IEEE 802,11 sera une des principales candidates.

Néanmoins, les conditions de communication pour les environnements industriels sont très spécifiques. Outre le trafic générique semblable au trafic trouvé dans des environnements bureautique/domestique, il existe le trafic temps-réel. Ce trafic est typiquement associé à des applications de contrôle, où les données de contrôle doivent être périodiquement transférées entre les capteurs, les contrôleurs et les actionneurs en conformité avec les spécifications temps-réel. En conséquence, une des questions fondamentales qui doivent être abordées, surtout dans le cas où les réseaux sans fil sont utilisés dans des environnements industriels est: "Comment garantir que les besoins temps-réel des données de contrôle sont respectés, quand le moyen de communication est partagé avec du trafic générique temporellement non borné?"

Le principal objectif de cette thèse est répondre à cette question. C'est-à-dire, développer de nouveaux mécanismes pour supporter des services de communication temps-réel sur une infrastructure de communications basée sur le

protocole CSMA, qui est aussi utilisé pour supporter du trafic générique temporellement non borné.

Cette thèse adresse les applications temps-réel, avec les besoins suivants: (i) environnement de communication avec des stations normalisées (stations ST) et un ensemble de stations temps-réel (stations TR), qui partagent le même domaine de communication; (ii) la charge imposée par les stations ST est temporellement non bornée, c'est-à-dire, elle est hors du contrôle de l'architecture TR. Supplémentairement, le suivant besoin est appliqué: Aucune modification matériel/logiciel doit être nécessaire à les postes ST dû à la grande quantité installée de dispositifs normalisés IEEE 802.3/802.11.

En conséquence, dans le contexte de cette thèse un nouveau moyen de communication TR (abordage VTP-CSMA) est proposé basée sur des mécanismes de séparation de trafic. Ces mécanismes sont capables de prioritiser le trafic TR par rapport à autres trafics, sans contrôler directement ces autres trafics. C'est-à-dire, au lieu de contrôler *tout* le trafic produit par *toutes* les stations, l'abordage VTP-CSMA contrôle seulement le trafic produit par les stations TR, de telle forme que la résolution de collisions est forcée à faveur des stations TR.

To all my family.

Acknowledgements

It is always hard to put down your feelings and acknowledgements in words. In my point of view, it is hard already in your mother tongue. Impossible in another language. I shall write it in Portuguese!

"O indivíduo realmente criativo está sempre pronto a abandonar as velhas classificações e a reconhecer que a vida, sobretudo a sua própria vida - absolutamente única - é rica de novas possibilidades".
(Frank Barron)

"Nada acontece a menos que sonhemos antes".
(Carl Sandburg)

A idéia deste trabalho nasceu em 2003, quando ≪sonhei≫ desenvolver os meus estudos de doutorado fora do Brasil. Projeto lançado, decisão tomada só me restava pôr o "pé na estrada". Deixar o país para a realização de um projeto como este é sempre uma possibilidade de construir novos aprendizados. Mas também é uma decisão que implica rupturas e distanciamentos: deixar a família, os amigos, os propósitos profissionais e pessoais e ir para uma outra realidade distante de tudo isto... Sentia que era chegado o momento certo!

Este caminho que fui trilhando esteve marcado pela grande saudade que sinto dos meus familiares e dos meus amigos que ficaram no Brasil. Diversos foram os acontecimentos que não pude presenciar, algumas foram as perdas que só me restaram vivê-las à distância. Por outro lado, foram também diversas as conquistas, as oportunidades e, sobretudo, foi grande o crescimento pessoal e profissional que pude construir. Assim, posso afirmar que tudo valeu a pena!

A conclusão deste trabalho é um momento muito especial para mim. Tenho a consciência de que algumas pessoas foram fundamentais para que ele se concretizasse. Por diferentes razões, eu gostaria de agradecer especialmente:

a Francisco Vasques, professor na Universidade do Porto – pelo exercício exemplar da sua função de orientador, pela acessibilidade e disponibilidade para ouvir com interesse todas as questões, dúvidas e problemas que surgiram no decorrer desta tese. Agradeço todos os desafios lançados ao longo desses 3 anos de trabalho, a confiança em mim depositada e, sobretudo, pelo apoio incondicional em todos os momentos. Tenho a certeza de que a realização deste projeto não teria sido possível sem o seu auxílio. Meus sinceros agradecimentos a você, Francisco, a quem dedico profunda e sincera admiração.

a Paulo Portugal, professor na Universidade do Porto – por todo o estímulo que recebi, pelas discussões científicas que tivemos a oportunidade de realizar, pelos resultados frutíferos colhidos ao longo deste período e que foram fundamentais para os resultados deste trabalho.

a José Alberto Fonseca, Pedro Souto e Stefano Vitturi, professores nas universidades de Aveiro, Porto e Pádova, respectivamente – pelo acompanhamento e pela excelente contribuição científica em parte dos trabalhos realizados no âmbito desta tese.

a Francisco Borges Carreiro, Paulo Bartolomeu e Valter Silva, da Universidade de Aveiro – pelo trabalho conjunto que realizamos.

a todos os funcionários da Faculdade de Engenharia, especialmente à Maria de Lourdes e à Maria Januária – pela alegria e disposição com que conduzem seus trabalhos e por todo o auxílio nas inúmeras vezes que necessitei.

a Max Mauro Silva, professor do Centro Universitário do Leste de Minas Gerais – pelo seu grande incentivo e pela ótima indicação do professor Francisco Vasques para ser o orientador desta tese: Muito obrigado!

"É preciso amar as pessoas como se não houvesse amanhã
Porque se você parar para pensar, na verdade não há".
(Dado Villa-Lobos, Renato Russo, Marcelo Bonfá)

a toda minha família – pelo apoio incondicional durante todo este tempo que estive ausente. Foi muito difícil passar esse período fisicamente longe de vocês. Porém, tenho a certeza que sempre estivemos juntos de alguma

forma. Esse trabalho não teria sido possível sem as constantes demonstrações de amor de todos vocês. O meu carinho e a minha gratidão especial à minha mãe, Tereza, por ensinar-me desde muito cedo a lutar pelos meus sonhos.

a Adilson e Bruno – por cada dia ao longo deste período, no qual compartilhamos muito mais que uma casa em Portugal. Agradeço a amizade, o carinho, a alegria e, principalmente, pela cumplicidade que construímos. Vocês contribuíram imensamente para que durante este período eu tivesse uma vida feliz em Portugal. É por isso que eu agradeço ter conhecido vocês... Enfim, agradeço pela oportunidade do estabelecimento de uma verdadeira ≪família d'alma≫.

a Ana Paula, Andreia e Reinaldo – pela amizade que construímos neste período. Ana Paula e Reinaldo: sinto que vocês são um só e espero que esta certeza possa torná-los ainda mais ≪Yin e Yang≫. Andreia: hoje eu vejo que o difícil não foi ≪aquele≫ primeiro final de semana no Porto. Mas, sim, todos os outros marcados por tua ausência. Obrigado por todos os ótimos momentos que passamos juntos. Conhecer vocês só reforçou em mim a certeza de que podemos até sobreviver sem algumas coisas, mas não sem amigos.

a Karin Chvatal, Kátia Ramos, Osvaldo Piedade Silva, Teresa Branco, Valério Rosset e Zenaide Alves – pela partilha de diferentes momentos, pela oportunidade de convívio e aprendizado com cada um de vocês.

a Clarice, Fatinha, Florinda, Sr. Francisco, Luís, Sr. Ribeiro e a todo o pessoal do "grupo" – pelo acolhimento neste espaço tão especial; pela dedicação para com o trabalho que vocês realizam e que eu aprendi a admirar. A presença de vocês e o aprendizado que me proporcionaram sempre contribuíram para o meu equilíbrio emocional e para o meu crescimento humano.

a Ana Barata e Cristina Loureiro, "minhas" professoras portuguesas de Inglês – por todo a aprendizado da língua inglesa ao longo deste tempo, pelas divertidas "aulas" que tivemos.

a Beto e Cheila – pelos verdadeiros laços de amizade que construímos nestes 13 anos de convívio.

a Alexandre Perin, Ailton, Analúcia, David, Jonnhy, Karina, Rosiléia, Valdeci e Viviane, amigos que conheci através das vivências acadêmicas na Universidade do Planalto Catarinense – pelo auxílio, incentivo e apoio em diversas fases da minha vida pessoal e profissional.

Por fim, agradeço à Fundação para a Ciência e a Tecnologia (FCT), ao Departamento de Engenharia Mecânica e Gestão Industrial (DEMEGI) e à Unidade de Integração de Sistemas e Processos Automatizados (UISPA) do Instituto de Engenharia Mecânica (IDMEC - polo FEUP) – pelo apoio financeiro e logístico neste estudo. Agradeço à Universidade do Planato Catarinense (Brasil) pela concessão da licença laboral para a realização deste projeto.

Porto, 12 de março de 2007.

Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Description
AC	Access Category
ACK	Acknowledgement
AIFS	Arbitration Interframe Space
AIFSN	Arbitration Interframe Space Number
AP	Access Point
ASIC	Application Specific Integrated Circuit
BB	Black-Burst
BEB	Binary Exponential Backoff
BER	Bit Error Rate
BLAM	Binary Logarithmic Arbitration Method
BK	Background
BS	Bucket Size
BSS	Basic Service Set
BTPS	Busy Tone Priority Scheduling
CABEB	Capture Avoidance Binary Exponential Backoff
CAP	Controlled Access Phase
CF	Contention Free
CFP	Contention Free Period
CIM	Computer Integrated Manufacturing
CIP	Control Information Protocol
COTS	Commercial Of-The-Shelf
CoV	Coefficient of Variation
CP	Contention Period
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CTS	Clear to Send

Acronym	Description
CW	Contention Window
DCF	Distributed Coordination Function
DCR	Deterministic Contention Resolution
DFS	Distributed Fair Scheduling
DIFS	Distributed Interframe Space
DMS	Delay Measurement System
EB	Energy-Burst
EBBS	Energy-burst for Bit Separation
ECs	Elementary Cycles
EDCA	Enhanced Distributed Channel Access
EDF	Earliest Deadline First
EIFS	Extended Interframe Space
EMI	Electromagnetic Interference
EQuB	Ethernet Quality of Service Using Black Bursts
EPSG	Ethernet Powerlink Standardization Group
ESS	Extended Service Set
FCFS	First-Come-First-Serve
FCS	Frame Check Sequence
FTDMA	Flexible TDMA
FTT	Flexible Time-Triggered
GDCF	Gentle DCF
HC	Hybrid Coordinator
HCCA	HCF Controlled Channel Access
HCF	Hybrid Coordination Function
HIMD	Harmonic-Increase and Multiplicative Decrease
IBSS	Independent BSS
IEEE	Institute of Electrical and Electronics Engineers
IEETA	Institute of Electronics and Telematics Engineering of Aveiro
IFS	Interframe Space
IP	Internet Protocol
ISO	International Organization for Standardization
LLC	Logical Link Control
LS	Latest Time
MAC	Medium Access Control
Mbps	Mega bits per second
MCU	MicroController unit
MLF	Minimum-Laxity-First
MP	Multipurpose
MSDU	MAC Service Data Unit
MSP	Message Stream Period

Acronym	Description
NAV	Network Allocation Vector
NCP	Network Control Packet
NCS	Networked Control Systems
NIC	Network Interface Card
ODVA	Open DeviceNet Vendors Association
OFDM	Orthogonal Frequency-Division Multiplexing
OSI	Open Systems Interconnection
PC	Point Coordinator
PCF	Point Coordination Function
PCSMA	Predictable CSMA
PDU	Protocol Data Unit
PHY	Physical
PIFS	Point Interframe Space
PLC	Programmable Logic Controller
QoS	Quality of Service
RP	Refresh Period
RT	Real-Time
RT-EP	Real-Time Ethernet Protocol
RTR	Ready-to-Receive
RTS	Request to Send
SCFQ	Self-Clocked Fair Queueing
SI	Service Interval
SIFS	Short Interframe Space
SoC	Start of Cycle
SPN	Stochastic Petri Net
TDMA	Time Division Multiple Access
TM	Trigger Message
TTP	Time-Triggered Protocol
TS	Traffic Stream
TSm	Traffic Separation mechanism
TSPEC	Traffic Specification
TXOP	Transmission Opportunity
UP	User Priority
VI	Video
VID	Virtual Identification
VLAN	Virtual Local Area Network
VO	Voice
VoIP	Voice over IP
VPL	Virtual Polling List
VPP	Virtual Polling Period

Acronym	Description
VPRP	Voice Packet Resolution Period
VTCSMA	Virtual Time CSMA
VTP	Virtual Token Passing
WLANs	Wireless Local Area Networks
WTRP	Wireless Token Ring Protocol
WTS	Wireless Traffic Smoother

Chapter 1

Overview

This thesis intends to be a contribution to the advance of the state-of-the-art in the context of real-time communications in CSMA-based networks. This chapter provides an overview of the research context of this thesis and its objectives. It also outlines the major contributions of this research work.

1.1 Introduction

The demand for real-time communication services has been increasing during the past few years. Driving examples are VoIP (voice over IP) and Networked Control System (NCS) applications. For such type of application domains, the support of reliable communications is one of the major requirements. For instance, in automation systems, real-time control data must be periodically transferred between sensors, controllers and actuators according to strict transfer deadlines. Real-time communication services are commonly classified according to the degree of real-time guarantees into the *hard* and *soft* real-time groups. Hard real-time applications require predictable and bounded response times, and any violation of these response times may have severe consequences (*e.g.* loss of human lives). Instead, soft real-time applications can tolerate some losses of temporal deadlines *e.g.*, real-time control application can tolerate occasional losses of the control law updates, especially if the control law has been modified to account for these lost updates [1]. However, real-time control applications are usually not resilient against jitter on the control law updates. In the case of a NCS,

it is of the utmost importance to have a nearly constant average communication delay and low jitter, whatever the behavior of the communication environment.

This thesis addresses network architectures and protocols which enable the support real-time communication services in broadcast random access networks. The major challenge concerning the design of protocol architectures for this type of networks is that the channel is a shared resource. Therefore, there is the need to prioritize real-time control data when such traffic shares the same communication infrastructure with generic multipurpose traffic [2]. Thus, access to this shared resource needs to be coordinated either centrally or in a distributed manner [3]. The Medium Access Control (MAC) protocols are the ones responsible for the access control.

Actually, the MAC protocol is the key issue in any broadcast random access network. Presently, the most used random access protocols in wired and wireless environments are based on IEEE 802.3 and IEEE 802.11 standard protocols[1]. There is a strong similarity between these two protocols. Both use the Carrier Sense Multiple Access (CSMA) mechanism to manage the medium access. The main drawback of both protocols is the nondeterminism of the probabilistic contention resolution algorithm that serializes the contending messages, whenever a collision occurs.

Ethernet is a well-known and extensively used network technology. The first standardized version was approved and released in 1985 as the ANSI/IEEE 802.3 standard [4]. When Ethernet networks started to be also used in the plant floor, simplicity, higher speed and low cost for the communication controllers were the major motivation [5]. However, the traditional shared Ethernet system, with its simple CSMA/CD (CSMA with collision detection) MAC protocol, do not easily allow the support of real-time (RT) communication services. Consequently, diverse commercial companies have developed extensions to the Ethernet standards to cover this problem. Today multiple systems have fulfilled the real-time Ethernet commercial specifications.

In the last few years, Ethernet has emerged as the *de facto* communication standard also for the lower levels of the industrial communication hierarchy[2]. In short, the real-time Ethernet is a fieldbus specification using

[1] The protocols defined by these standards are also known as Ethernet and WiFi protocols, respectively.

[2] The industrial or factory floor communication systems will be extensively used in this thesis as an example of real-time communication systems.

Ethernet for the two lower layers [6]. Thus, industrial real-time devices cannot be as cheap as in office environments, where costs of standard devices have been optimized in large scale production. As a consequence, there are still many research projects proposing solutions addressing RT communication support in Ethernet-based networks.

The IEEE 802.11 family of protocols is one of the most used sets of Wireless Local Area Networks (WLANs). It was standardized in 1999 by the IEEE as the IEEE 802.11 standard, which was later reaffirmed in 2003 [7]. Recently, the IEEE 802.11e [8] standard has been published as an amendment to the original standard. This amendment is intended to provide differentiated levels of Quality of Service (QoS) to the supported applications, including the transport of voice and video over WLANs.

The demand for high performance industrial wireless networking will increase significantly in the next few years. This is a consequence of recent technology developments that demand wireless access in office environments, in public hot-spots and in domestic environments. Actually, nowadays industrial environments are already making big efforts to move from wired to wireless networks [9]. Therefore, it is reasonable to expect that in the near future, the widespread availability of wireless solutions will generate a similar *de facto* standard for industrial wireless communications.

Within this context, the IEEE 802.11 family of protocols is one of the main contenders to become the *de facto* standard for industrial wireless communications. One of the main reasons for is that this family of protocols is easily able to replace Ethernet in a transparent way, implementing the two lowest layers of the ISO/OSI model, the IEEE 802.11 protocol provides all the required functionalities to enable the support of the Internet Protocol (IP) that is virtually the basis for applications over Ethernet networks.

Traditionally, the RT communication behavior in wired CSMA environments has been guaranteed through the tight control of every communicating device [5]. The coexistence of RT controlled stations with timing unconstrained stations has been made possible by constraining the traffic behavior of the latter. For instance, using traffic smoothers. Unfortunately, when moving from wired to wireless networks, this traffic smoothing *paradigm* is no longer adequate, as it is not possible to impose any traffic smoothing strategy upon stations that are out of the sphere-of-control[3] of the RT architecture. The main reason is that the wireless physical medium

[3] The concept "inside/outside" sphere-of-control was defined by Kopetz [10]. We are using this term in conformity with Kopetz's definition. Whenever a RT entity is in the

is a relatively[4] *open communication environment*. That is, any new participant can try to access the communication medium at any instant (according to the MAC rules) and establish its own communication channels. Furthermore, the random bit error rate (BER) cannot be considered negligible in wireless networks. The wireless communication environment is susceptible to interferences created by other systems, not only from those using the same technology, but also from other technologies working in the same frequency band [11]. As a consequence, the system load cannot be predicted at system setup time, nor can it be effectively controlled during the system run-time.

It is foreseeable that the existing RT networks will be challenged for moving from *closed* to *open* communication environments, and also for partially moving to a wireless network infrastructure. Thus a *new paradigm* for real-time communications will need to emerge, as the traditional RT communication paradigm is still based on closed and controlled environments. Throughout this thesis, it will become clear that the use of WLANs in real-time communication domains will challenge the referred RT communication paradigm. More accurately, we have identified that the most promising solutions to provide RT communication in the next generation of communication environments will be those that allow the coexistence of both real-time and non real-time stations in the same communication domain, enabling the prioritization of the real-time traffic without the need to control every communicating device.

1.2 Research Context

In this thesis, we investigate the use of novel solutions to provide RT communication in both wired Ethernet and wireless WiFi networks according to an open communication *paradigm*. The use of this open communication paradigm, especially when considering wireless network infrastructures, reopens several RT challenges that have to be re-addressed. The most important ones are: how to guarantee the timing requirements in an error-prone

sphere-of-control of a subsystem, it belongs to a subsystem that has the authority to change all the value of this RT entity. Outside its sphere-of-control, the value of the entity can be observed, but cannot be modified.

[4] In this work, the term "relatively" is used mainly because the "security aspects" of communication are out of the scope of this thesis.

channel subject to interferences, and how to guarantee the timing requirements of RT channels when the communication medium is shared with timing unconstrained traffic.

Nowadays, there are several approaches and techniques that commonly provide real-time behavior to Ethernet-supported applications. However, few of those techniques allow enhanced (real-time) stations[5] to coexist with standard (non-real-time) stations in the same network domain, without strictly controlling the traffic behavior of the latter. Some of those proposals have been adapted to WLANs and there are also multiple innovative approaches being developed to provide real-time behavior to wireless-supported applications. However, the majority of these new mechanisms still do not consider that the wireless physical medium is a relatively open communication environment, where any new station can start trying to transfer its own traffic according to the medium access rules.

Therefore, one of the fundamental questions addressed in this thesis is: "How to guarantee the timing requirements of RT applications, when the communication medium is shared with timing unconstrained multipurpose traffic?" Within this context, this thesis investigates network architectures intended to provide RT communication in CSMA-based networks, where a special emphasis is given to solutions compatible with both IEEE 802.3 and IEEE 802.11 standards.

In this thesis, simulation models have been used to assess the real-time behavior of the analyzed communication proposals. It is worth mentioning that simulation-related metrics are all based on the average behavior (average delay, average queue size, etc.). It is well known that the analysis of the real-time behavior should have been made for the worst-case scenarios. However, worst-case scenarios in probabilistic medium access networks address rarely occurring cases and, those rarely occurring cases may only be relevant for safety-critical applications.

Therefore, most part of the research work presented in this thesis has been focused on the average behavior. The main reason for this option is that the target applications are usually loss tolerant in what concerns the lost of some message deadlines. For instance, the transfer of a video stream may be specified to tolerate a maximum of 10% deadline loss rate, if the lost frames are "adequately" spaced. Another examples of relevant

[5] The terms "stations", "nodes" and "devices" are used with the same meaning in this thesis. They represent any communication entity with communication capability via wired or wireless channels.

loss tolerant applications are NCSs scheduled according to the (m,k)-firm model [12], or the support of VoIP applications, where an average packet delay below $150ms$ and an average jitter $< 50ms$ are acceptable for most user applications [13].

1.3 Research Objectives

The main objective of this thesis is to research and develop novel solutions to support RT communication in IEEE 802.3 and IEEE 802.11 communication networks, specially when these networks are also being used to support timing unconstrained traffic. This thesis will focus in the evaluation of the RT characteristics of the IEEE 802.3 and IEEE 802.11 protocols and it will propose new RT communication mechanisms based on the use of specifically designed traffic separation mechanisms.

The fundamental assertion of this thesis is that *traffic separation mechanisms* are adequate mechanisms to support real-time communication in open communication environments. That is, in communication environments where the system load cannot be predicted at system setup time, nor can be effectively controlled during the system run-time.

1.4 Main Contributions of this Thesis

The main contributions of this thesis are:

1. The specification of a new Traffic Separation mechanism (TSm) for CSMA networks. The TSm mechanism allows the coexistence of IEEE 802.3/802.11 standard stations with modified (real-time) stations in the same network domain, prioritizing the real-time traffic (Moraes *et al.* [14]; Moraes *et al.* [15]; Moraes and Vasques [16]; Moraes and Vasques [17]);

2. The specification of the VTPE-hBEB architecture, which is a shared Ethernet deterministic architecture. Such specification involves a temporal analysis of the proposed protocol and its experimental assessment made within the context of a collaborative project with the University of Aveiro (Carreiro *et al.* [18]);

3. The specification of the VTP-CSMA architecture, which is a wireless RT architecture that considers an unified wireless system in one frequency band, within which the bandwidth is shared by real-time and non real-time communicating stations (Moraes *et al.* [19]; Moraes *et al.* [20]);

4. The specification of an adequate ring management procedure that allows stations to dynamically leave or join (rejoin) the VTP-CSMA architecture (Moraes *et al.* [21]);

5. The assessment of the suitability of the IEEE 802.11e EDCA standard mechanism to support real-time communication in the next generation communication environments (Moraes *et al.* [22]);

6. The implementation of a Stochastic Petri Net (SPN) simulation model that describes the dynamics of the Contention-Based Channel Access Function (EDCA) of the Hybrid Coordination Function (HCF) of IEEE 802.11e amendment. This model was used for the performance analysis of the both EDCA and VTP-CSMA protocols (Moraes *et al.* [23]).

1.5 Thesis Outline

The remainder of this thesis is organized as follows. In **chapter 2**, some of the most relevant approaches to support real-time communication in CSMA-based networks are surveyed. This chapter starts by presenting the state-of-the-art on RT communication in IEEE 802.3 wired networks, followed by the state-of-the-art on RT communication in IEEE 802.11 wireless networks. A classification framework structured in two classification axes is proposed. The first axis is related to how collision are dealt with, in order to provide RT communication services. Such state-of-the-art approaches are classified in three groups: avoiding collisions; solving collisions or; reducing collisions. Finally, a synthesis is made, where the described state-of-the-art approaches are classified according to a second classification axis related to its compatibility degree. Besides, throughout this chapter are identified the most promising CSMA-based approaches that will enable the support of real-time communication in the next generation communication environments.

Chapter 3 includes an overview about how the CSMA protocol works, for the case of collision detection (CSMA/CD) and collision avoidance (CSMA/CA). A special emphasis is given to the additional EDCA coordination function incorporated by the recent IEEE 802.11e amendment. Then a new Traffic Separation mechanism (TSm) that prioritizes real-time traffic over unconstrained traffic is presented. This mechanism is intended to be used as an underlying traffic separation mechanism, that will enable the provision of real-time communication services in CSMA-based networks. Therefore, simulation and analytical studies are extensively discussed.

Chapter 4 presents and evaluates an innovative shared Ethernet deterministic architecture able to interconnect sensors, controllers and actuators at the field level of an industrial communication architecture. The proposed VTPE-hBEB architecture is presented in detail from a functional point of view. Furthermore, a timing analysis complemented by some experimental results are presented. Such experimental results have been developed in cooperation with Institute of Electronics and Telematics Engineering of Aveiro (IEETA) at University of Aveiro. Additionally, this chapter presents a brief state-of-the-art on real-time Industrial Ethernet, where some of the most used Industrial Ethernet solutions are described.

Chapter 5 assesses the timing behavior of the EDCA mechanism of the IEEE 802.11e standard communication protocol, when it is used to support real-time traffic. Basically, it is assessed the behavior of the voice category in open communication environments, when this access category is used to transfer small sized packets, generated in periodic intervals. The goal of this chapter is to provide a thorough understanding of the limitations of the EDCA mechanism, when it is used to support RT traffic in open communication environments.

Chapter 6 presents a wireless real-time communication architecture, based on a virtual token passing procedure. This architecture, referred as VTP-CSMA architecture considers an unified wireless system operating in one frequency band, with the communication bandwidth shared by real-time and non real-time communicating devices. The VTP-CSMA architecture is the main contribution of this thesis. This chapter shows that the VTP-CSMA architecture enables the real-time communication support in unconstrained environments, in the presence of a variable number of real-time stations.

Finally, **Chapter 7** discusses the conclusions and presents future possible directions that may emerge from this work.

Chapter 2

Review of Relevant Work

The purpose of this chapter is to survey the state-of-the-art on real-time communication in CSMA-based networks. A special emphasis is given to IEEE standard wired and wireless networks, which operate according to the CSMA/CD or CSMA/CA protocols in shared broadcast environments. The purpose of this chapter is also to identify the most promising solutions to support real-time communication in the next generation communication environments, characterized by a communication medium shared with timing unconstrained devices that generate an unpredictable network load.

2.1 Introduction

The Carrier Sense Multiple Access (CSMA) is a well-known and extensively used Medium Access Control (MAC) protocol. Its inherent simplicity has imposed it as one of the preferred solutions to implement communication protocols in shared broadcast networks. Relevant solutions based on the CSMA protocol are the IEEE 802.3 [24] and 802.11 [7] standards, also known as Ethernet and WiFi protocols, respectively. The most relevant aspects of these two protocols will be detailed in chapter 3.

The referred simplicity is one of the main reasons for the success of CSMA-based networks. Such simplicity derives from the collision detection/avoidance mechanism that operates whenever randomly initiated transmissions collide. Whenever a collision is "detected", a distributed

probabilistic algorithm tries to serialize the contending messages. Such distributed algorithm is based on the local knowledge of the occurred collisions. One of the main disadvantages of such an approach is the non-determinism of the probabilistic contention resolution.

Supporting real-time communication in CSMA networks is a hard task. Traditionally, in wired environments, the real-time (RT) communication behavior is guaranteed through the strict control of every communicating device. State-of-the-art RT communication approaches range from the modification of the MAC layer, to the micro segmentation of the network. For some of the proposed approaches, the coexistence of RT controlled stations together with timing unconstrained stations is made possible by constraining the traffic behavior of the latter. Unfortunately, such type of approaches are not adequate for wireless environments, since it is not possible to impose any traffic constraining strategy upon *third* stations that may unpredictably access the communication medium.

This chapter reviews the state-of-the-art on real-time CSMA communications, in both wired and wireless network environments. A classification framework structured in two classification axes is proposed. The first axis is related to *how collisions are dealt with* in order to provide a RT communication service. Traditional approaches follow an *avoiding collision* strategy to guarantee a RT communication service to the supported applications. Another possibility is to replace the traditional probabilistic collision resolution algorithm by an algorithm that ensures an adequate *deterministic collision resolution*. Finally, it is also possible to enforce the *reduction of the number of occurring collisions*, through the use of adequate loosely-coupled distributed algorithms.

The proposed classification framework is complemented with a second classification axis, that is related to the *compatibility degree* with IEEE standard devices. Specifically, this axis highlights how the proposed RT communication approaches keep or alter the compatibility with IEEE 802.3/802.11 compliant devices. Three different compatibility levels have been defined. The *level 1* subclass gets together RT communication proposals requiring the compliance of *all* communicating devices (both real-time and non real-time) with the enhanced (real-time) devices. Such enhancements can range from small modifications at hardware/firmware level, to the use of a fully modified MAC protocol on top of IEEE 802.3/802.11 physical layers. Therefore, the main characteristic of compatibility level 1 is the impossibility of coexistence between enhanced (real-time) devices and default devices in the

same network domain. On the other hand, *level 2* and *level 3* subclasses comprises RT communication proposals able to offer RT guarantees in presence of *third* devices. The main difference between these two subclasses is related to the required level of modifications. The implementation of a level 2 device requires the use of specific hardware, impairing the use of COTS (commercial off-the-shelf) hardware. Conversely, a level 3 device can be implemented upon COTS hardware, requiring just modifications at the firmware/software level of the *real-time* communicating devices. This is an important distinction, as the possibility of using COTS hardware is a relevant advantage when setting-up a RT communication infrastructure.

The remainder of this chapter is organized as follows. Section 2.2 describes the state-of-the-art on RT communication in IEEE 802.3 wired networks, according to the first classification axis. Similarly, Section 2.3 describes the state-of-the-art on RT communication in IEEE 802.11 wireless networks. Afterwards, in Section 2.4 a synthesis of the state-of-the-art is made, where the described research proposals are classified according to the second classification axis. One of the main purposes of Section 2.4 is to identify the most promising real-time CSMA-based approaches to handle the requirements imposed by the next generation communication environments. Specifically, the requirements imposed by a communication medium shared with timing unconstrained devices, generating an unpredictable network load. Finally, a summary of this chapter is drawn in Section 2.5.

2.2 RT Communication in IEEE 802.3 Wired Networks

The IEEE 802.3 is a widely used network technology, with a Medium Access Control protocol based on the Collision Detection among randomly initiated transmissions (CSMA/CD). It has a non-deterministic behavior, due to the use of a probabilistic contention algorithm, where the retransmission probability does not depend on the type of traffic, but just on the state of the collision counter of each particular station.

As a consequence, it impairs the support of real-time communications. Nevertheless, multiple approaches and techniques have been developed to provide real-time communication services to IEEE 802.3 supported applications. In this subsection, a number of relevant research solutions addressing this issue are reviewed.

2.2.1 Avoiding collisions

The first approach that can be considered when dealing with collisions is to avoid its occurrence. The most popular solutions to *avoid collisions* are based on the TDMA (Time Division Multiple Access) paradigm. TDMA is a channel access method, which allows multiple nodes to share the same channel by dividing the access time into different time slots. Messages are sent at exclusive time slots assigned to each node. It implies a precise clock synchronization among the different nodes so that all nodes are able to agree on their respective transmission slot. One example of a TDMA real-time network was proposed by Chen and Lu [25]. This protocol dynamically combines CSMA/CD with a modified version of TDMA, where in the called "under control" state, a NCP (Network Control Packet) message is broadcasted by the control station to synchronize all other stations and to inform them that a new frame started.

A similar approach was proposed by Pritty *et al.* [26], based on the use of the Timed Packet Release principle, where a monitor node periodically transmits a slot pulse to synchronize the medium access. The monitor node is located at one end of the bus. For each station is allocated a unique time delay, which starts from receipt of the slot pulse. Stations can only start their transmission at the expiration of its time delays, if the bus is silent at that time. Therefore, the value of the time delay assigned to each station is of paramount importance. Naturally, the smaller delay value leads to the higher priority. However, the increment in the value from station to station must be enough to ensure unambiguous detection of the start of a packet transmitted from a previous node.

The TTP (Time-Triggered protocol) [27] is a well-known protocol based on the TDMA paradigm. This protocol has been developed at Technical University of Vienna to be applied in safety critical real-time systems. The current version (TTP/C) is built upon the IEEE 802.3 physical layer [24]. In TTP/C, messages are sent in TDMA rounds, where every node must send a message in every round. Communication on the network is based on static tables loaded at each node. A general purpose message may contain up to 240 bytes of data. Frames are transmitted at speeds up to 25 Mbps with typical data efficiency of 60%. The most simple configuration of a TTP system is shown in Figure 2.1, where each controller has two bidirectional communication ports, each one connected to a given TTP replicated channel.

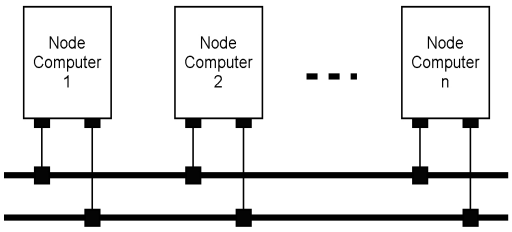

Figure 2.1: A basic topology of TTP/C-bus.

Some recent solutions to *avoid collisions* use Master-Slave techniques, where a special node, the *master*, instructs the other nodes, the *slaves*, to transmit at specific instants. Pedreiras *et al.* [28] proposed a master/multi-slave transmission control technique called FTT-Ethernet (Flexible Time-Triggered Ethernet) to schedule communications in a shared Ethernet networks. The FTT-Ethernet approach provides deterministic access to all stations, according to a specifically defined reservation technique. In this approach, time is divided in synchronous and asynchronous windows, which are used to, respectively, statically schedule the hard real-time traffic and dynamically serve the soft real-time requests (control messages, event-triggered messages and non-real-time traffic).

Basically, the master node implements a centralized scheduling concept, in which both the communication requirements, message scheduling policy and on-line admission control are localized in one single node. The distribution of the scheduling decisions to the network stations is periodically performed by the master through a special control message, the trigger message (TM). Besides, the FTT-Ethernet paradigm uses a technique, in which a single trigger message causes the transmission of multiple slaves messages, eventually originated in distinct station nodes. Obviously, this method reduces the number of control messages, improving the bandwidth utilization [29]. Figure 2.2 illustrates the FTT-Ethernet paradigm, where the bus time is slotted into consecutive fixed duration time-slots, called Elementary Cycles (ECs) and the master issues one control message only, indicating which data message must be transmitted therein.

Ethernet Powerlink [30] is a commercial master-slave protocol based on the standard IEEE 802.3 layers. Deterministic time is achieved by applying a cyclic timing schedule to all the connected nodes. Each cycle is divided in four distinct phases: start, cyclic, asynchronous and idle periods (Figure 2.3).

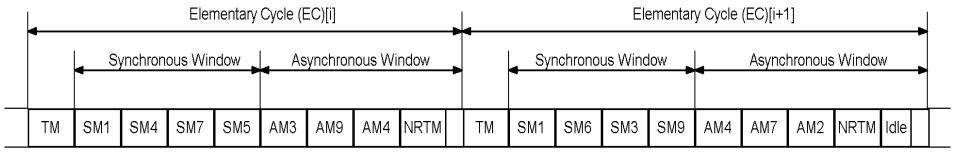

Figure 2.2: The FTT-Ethernet traffic structure.

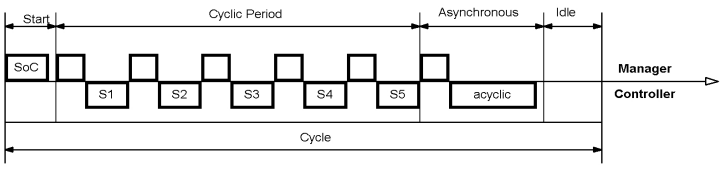

Figure 2.3: Ethernet Powerlink cycle.

In the start phase, the node management grants the access to the physical medium via the exchange of an explicit frame (SoC - start of cycle) transmitted as a broadcast message to all controllers, thereby preventing collisions. During the cyclic period, time-critical data is transferred through a configurable schedule scheme, where the manager transmits a "Poll Request" frame to each controller. Upon reception of a "Poll Request", the controller responds by transmitting the correspondent data message. The asynchronous phase reserves bandwidth for non time-critical data. The Idle period represents the unused period until the new Ethernet Powerlink cycle begins. Any topology can be implemented using hubs. Despite of a recent version (version 2.0) that also allows operation over Switched Ethernet networks, the Ethernet Powerlink Standardization Group (EPSG) recommends the use of repeater hubs instead of switching hubs within the real-time domains, to minimize path delays and frame jitter.

The Token Passing is other well-known method to *avoid collisions* in shared broadcast bus networks. This method consists in circulating a token among the stations, where each station is allowed to access the medium only during the token holding intervals. The timed-token protocol [31] is the basis for the RETHER protocol, which was proposed by Venkatramani and Chiueh [32]. This protocol operates in normal Ethernet CSMA/CD mode until a real-time request arrives. Then, it switches to the RETHER mode, where all nodes operate according to a token passing protocol. In this mode, time is divided into cycles during which the token regulates the bus

access. The token will firstly serve all the real-time nodes and then, if there is still time left during the cycle, it will also visit the non-real time nodes. Additionally, every new real-time request goes through an admission control system that determines if the request can be satisfied without affecting the existing schedule.

The RT-EP (Real-Time Ethernet Protocol) [33] is also based on an explicit token-passing procedure, where the access to the bus is carried out in two phases: *arbitration* and *application* message transmission. In the first, the token visits all the nodes to determine the one holding the highest priority message ready to be transmitted. Then, the token is sent directly to the node having the highest priority message for transmission (application message transmission phase). After concluding the application message transmission, the same node starts a new arbitration phase. This means that there is the need for one complete token rotation for the transfer of each single message.

In [34], J. Lee *et al.* proposed the use of the IEEE 802.4 Token-Passing Bus Access method [35] directly on top of the Ethernet Physical Layer, where a specifically proposed service translator performs the required translation of frame formats and interface functions.

Finally, the most recent solution to *avoid collisions* is based on the micro segmentation of the network, using the IEEE 802.1 Switched Ethernet standard [36]. This standard that was introduced in the early 1990s, enables the micro-segmentation of the network by regenerating information only to the receiving port of the Switch. When using Switched Ethernet, it is possible to manage network traffic, by means of the adequate setting of data flow permissions and priorities. The transfer of critical information was addressed both by the IEEE 802.1p and the IEEE 802.1q VLAN [37] standards; the latter extends the priority handling aspects of the 802.1p standard, by providing space in the VLAN (Virtual Local Area Networks) Tag to indicate traffic priorities to support VLANs, while the former gives the ability to prioritize messages.

Nevertheless, the use of switches in an Ethernet network is not a panacea. For instance, if the traffic is sent to an output port at a higher rate than its capacity, messages must be queued. If queuing occurs in an uncontrolled way, the switch can lose messages. Another important problem concerning the use of Switched Ethernet is the lack of enough priority levels to support efficient priority-based scheduling [38]. The impact of network topology and message scheduling strategies inside the switch has also been recently

addressed [39]. Moreover, switches induce higher delay and jitter in message forwarding than the traditional shared Ethernet mode (using hubs) [40]. Therefore, the hard real-time guarantees can only be achieved by the use of adequate admission control techniques.

It is worth noting that Switch-based communications are no longer related to the use of a CSMA-based MAC. Instead, they use a centralized communication scheduler: the Switch. Multiple improvements to the Switched Ethernet approach have been suggested, mainly to eliminate some of its technological problems [39, 41, 42]. Nevertheless, these solutions are out of the scope of this survey, as they are not related to the use of a CSMA medium access control.

2.2.2 Deterministic collision resolution

The second approach to support RT communications in CSMA-based networks is based upon the *deterministic collision resolution*. This can be achieved using one of two different approaches. The first one is by *modifying the collision resolution* algorithm, in order to guarantee that the colliding frames are serialized in an upper-bounded time interval. The second one is *forcing the collision resolution* in favor of the RT station, compelling all the other contending stations to abandon the medium access.

In the context of the first approach (*modifying the collision resolution algorithm*), Takagi *et al.* [43] proposed a CSMA/CD protocol with deterministic contention resolution (DCR). In the absence of collisions, the CSMA/DCR protocol implements the CSMA/CD access method. Whenever a collision occurs, a binary search tree is used to sort the colliding nodes. A priority hierarchy is enforced, *i.e.*, higher priority nodes try to access the medium prior to the lower priority ones, using an implicit token passing mechanism. Basically, when a collision occurs, the collision period is broken into a previously defined number of time slots. Then, each station can only transmit during its own slot. If a station detects a transmission, it interrupts the slot counting until the end of the transmission. Afterwards, the station waits another interframe space and the slot counting resumes. Consequently, when multiple stations want to transmit, they have to wait for their assigned slot.

Figure 2.4 together with Table 2.1 shows an example of CSMA/DCR execution on a binary tree, in a network with 8 stations, where 4 messages

collide. The active nodes with ready messages to be transmitted are represented in dark circles. After the initial collision, only nodes belonging to the left subtree transmit. The right branch of the tree (stations 6 and 8) ceases contending for the bus access. Since there are still two messages on the left branch, a new collision occurs, between nodes 3 and 4. Thus, the left subbranch is selected again and allows only nodes 1 and 2 to transmit. Since none of them have a message to send, the slot is empty (denoted by E). Then in the fourth slot, stations 3 and 4 will collide again. Finally, in the following slot, only the message from station 3 is selected, resulting in a successful transmission. The algorithm continues this way until all messages are successfully transmitted. Overall, this example requires 9 slots.

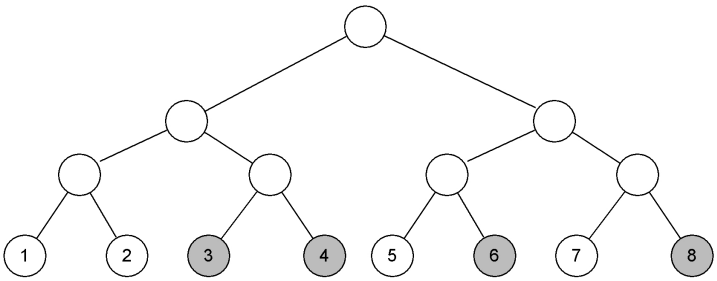

Figure 2.4: Example of tree search - CSMA/DCR.

Table 2.1: Tree Search Example - CSMA/DCR.

Slots	1	2	3	4	5	6	7	8	9
Status	C	C	E	C	T	T	C	T	T
Station	3 4 6 8	3 4		3 4	3	4	6 8	6	8

Note: Status (C-collision; E-empty; T-transmission).

The DOD-CSMA-CD [44] protocol improved the CSMA/DCR protocol, using network station indexes computed on-line, rather than pre-assigned ones. The dynamically computed indices are called time indices. They represent deadline equivalence classes. All messages that are assigned the same index have comparable deadlines. This protocol implements a scheduling algorithm similar to the Earliest Deadline First algorithm (EDF) [45].

In the context of the second approach (*forcing the collision resolution*), one of the first proposals has been made by Gopal and Wong [46]. The hybrid control token-CSMA/CD protocol has a behavior that is similar to

the VTPE-hBEB architecture proposed in this thesis (chapter 4). In hybrid control token-CSMA/CD proposal, all stations except the one holding the token work according to the standard CSMA/CD protocol, *i.e.*, a station having a packet ready to be transmitted, senses the channel and transmits the packet if the channel is sensed idle. If a collision occurs all involved stations terminate their transmissions and schedule retransmissions after a random backoff interval. Conversely, the station that possesses the token gets a higher priority over others. Specifically, when there is a collision on the channel, the station with the token continues to send data until all other stations are in their backoff phase. Then, the token holding station retransmits its packet immediately (without any backoff interval). The token is passed to the next station implicitly, each time the channel state change from busy to idle. The hybrid control token-CSMA/CD protocol requires specific hardware to be implemented, as it does not comply with the timing behavior defined for IEEE 802.3 standard devices. Conversely, the VTPE-hBEB architecture does not suffer from this limitation.

Sobrinho and Krishnakumar in [47] designed the EQuB (Ethernet Quality of Service Using Black Bursts) mechanism that also prioritizes the real-time traffic by forcing the collision resolution in favor of the RT station. EQuB allows the coexistence of real-time and non-real-time traffic on the same network domain, providing a bounded delay to real-time packets. The EQuB mechanism is based on the assumption that real-time stations generate packets in specific intervals, designated as *sessions*. During a session, real-time stations expect to have undisputed access to the medium through the EQuB mechanism.

The EQuB mechanism works as follows. At the beginning of a session, a real-time station works with the conventional BEB (Binary Exponential backoff) algorithm (according to the Ethernet rules). However, whenever real-time stations participate in a collision, they transmit a jamming sequence up to a pre-specified maximum allowed interval of time, in order to avoid that other stations acquire the channel. The jamming signal is called a *black burst*. The maximum duration of a black burst is a direct function of its contention delay, measured from the time where the access attempt has been scheduled until the host perceives the medium to be idle during an IFS. During the transmission of its black burst, a station continuously monitor the channel. If the station detects that no other stations are sending black bursts, it immediately re-initiates the transmission of its packet with success. On the other hand, if the station exhausts its black burst trans-

mission and still feels the bus jammed, it waits for the channel to become idle again (during an IFS) and repeats the algorithm.

Yavatkar *et al.* [48] proposed the PCSMA (Predictable Carrier Sense Multiple Access) protocol, which is another proposal that follows the paradigm of *forcing the collision resolution* in favor of the RT station. The PCSMA uses reservation and persistence to obtain predictable performance, where the periodic source must reserve the necessary transmission bandwidth before beginning its transmission.

Under PCSMA, when a periodic source (real-time stations) collides with a datagram source (non real-time stations), it does not backoff, but instead continues to transmit. A periodic packet contains enough overhead bits so that a periodic source does not transmit useful data until the collision with the datagram source(s) is resolved. Figure 2.5 shows the structure of a frame transmitted by a periodic source.

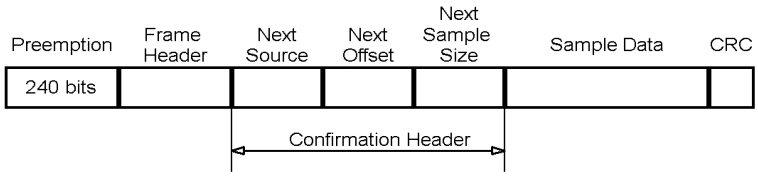

Figure 2.5: The structure of periodic frames.

In order to avoid collisions between two real-time transmissions, a periodic source generates and transmits a frame to reserve the transmission slots before starting the transmission. If two reservations conflict, one of the two sources must delay its transmission by a slot or more, since the delivery deadlines of both sources are met. Thus, periodic sources do not collide with each other.

2.2.3 Reducing the number of occurring collisions

Finally, the third approach to support RT communication in CSMA-based environments aims to *reduce the number of occurring collisions*, in order to enhance the network responsiveness to real-time message requests. Within this context, Molle and Kleinrock [49] proposed the Virtual Time CSMA (VTCSMA) algorithm. It uses a probabilistic approach combined with specific timing parameters (arrival time, laxity[1], deadline, length) for the col-

[1] time to the deadline minus message transmission time.

lision resolution. The VTCSMA protocol works as follows. Each station maintains two clocks: a real-time clock and a virtual time clock. The virtual clock runs faster than the real-time one. The virtual clock stops running when the channel is busy, and runs when the channel is idle. The original proposal uses the arrival time to determine when to transmit a message. In this case a message is sent only when its arrival time is equal to the time of the virtual clock.

However, this protocol allows implementing multiple scheduling policies by assigning different waiting times to pending messages. Zhao and Ramamritham [50] presented a performance analysis of the four VTCSMA protocols: VTCSMA-A, -T, -D and -L. The VTCSMA-A implements the Molle's original VTCSMA. The VTCSMA-T runs the virtual clock along the message length axis, where the message with the minimum length is transmitted first. The VTCSMA-D implements the minimum-deadline-first transmission policy and, the VTCSMA-L uses message laxity to determine when to transmit a message. The authors concluded that when the load is very light, for all four protocols, collisions are extremely rare. Other relevant result is that the VTCSMA-D achieves the best performance in terms of message loss and collision channel utilization.

Figure 2.6 show an example of the VTCSMA-D protocol. It considers 4 nodes connected to a shared Ethernet bus. The message parameters are shown in Table 2.2. D and L are the deadline and laxity of the message, respectively. It is assumed that each message have a transmission time equal to 14 time units and the propagation delay equal to 1 time unit.

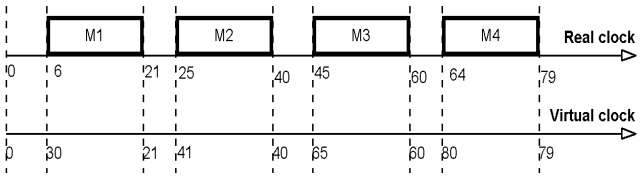

Figure 2.6: Example of VTPCSMA-D protocol.

From Table 2.2, each node has a message to be transmitted, labelled M1, M2, M3 and M4. As mentioned before, the virtual clock runs at a higher rate that the real-time one. If $\eta = 5$, the virtual clock runs 5 times faster than the real-time clock. For message M1, the deadline is 30. Then, when the virtual clock reaches 30, message M1 is transmitted. This virtual

Table 2.2: Real-time message parameters.

Node	M	Arrival time	D	L
1	1	0	30	15
2	2	10	41	26
3	3	20	65	50
4	4	20	80	65

time corresponds to real time 6. The message takes 15 time units to be transmitted. During this transmission, the virtual clock freezes. At the end of the M1 transmission, the virtual clock is reset to the real-time clock value, which is 21 and becomes active again. Therefore, M1 will meet its deadline. Afterwards, when the virtual clock reaches 41, message M2 is transmitted. This virtual time will correspond to the real time 25. Again, the message will meet its deadline and, messages M3 and M4 will be also successfully transmitted meeting their deadlines. Other relevant proposals implementing scheduling policies applied to Virtual Windows can be found in [51, 52].

The Window Protocol [53] is another relevant proposal, which implements a dynamic time window to reduce the number of occurring collisions. It operates as follows: all stations continuously monitor the channel, and after every successful message transmission, all stations select an initial *time window*. Once an initial *time window* has been selected, if just one station has a message ready to be transmitted and the message is within the window, then it will be sent. If several stations have messages to be transmitted within the window, the window size is reduced until there is just one remaining message within the window; if there are no nodes with messages within the window, the window size can be increased.

In the Window protocol all stations must follow the same policy to select the initial *time window*, as well as to update the *time window* whenever a collision occurs. In [54, 55], Kurose *et al.* proposed a Window protocol implementing the minimum-laxity-first (MLF) policy. They assume that the laxity of all messages are constant. Zhao *et al.* [50] suggested a Window protocol implementing the latest time to send policy (LS). Furthermore, a newly arriving message is immediately considered for transmission, if its LS is smaller than those of all pending messages in the system.

Figure 2.7 shows an example of the Zhao's proposal, where messages A and B are in the system at $t = 0$. Message A (MA) has its LS equal to 4 and length 1, and message B (MB) has its LS equal to 16 and length

2. The initial window size has been chosen to be 20. At $t = 0$, MA and MB are transmitting over the channel, causing a collision. At $t = 1$, the collision is detected and the transmission is aborted. At $t = 2$, the channel is idle and the window is reduced. The new window has been chosen to be 11 ($t \leq LS_M < 11$). As MA is within the window, it is transmitted. At $t = 3$, MC arrives in the system and, at $t = 4$ the channel is idle and the window goes back to the initial value ($t \leq LS_M < 20$). As both MC and MB are in the window, they start to be transmitted, causing a collision. Consequently, only at $t = 6$, the channel becomes idle and the window size is reduced to 13 ($t \leq LS_M < 13$). Then MC is transmitted and finally, MB will be transmitted.

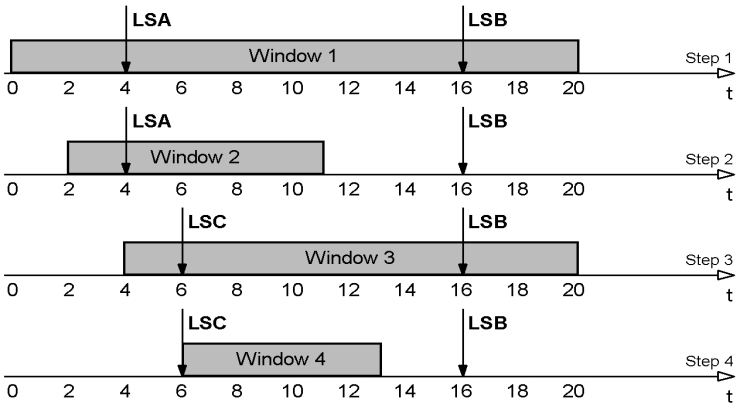

Figure 2.7: Example of Window protocol.

In [56], the authors presented a modified CSMA/CD protocol, called the Dynamic pi-persistent CSMA/CD protocol. The pi-persistent protocol is a variant of the window protocol, where the transmission probability of a ready packet depends on its laxity, and a time window is used to reduce the number of collisions in heavily loaded scenarios.

Molle *et al.* [57] proposed a BEB compatible algorithm, the Binary Logarithmic Arbitration Method (BLAM), that was studied by the 802.3w working committee as a means to solve the *Packet Starvation Effect*. Whetten *et al.* [58] demonstrated that, in heavily loaded networks, an older packet will have a smaller probability to be transferred than a newer one. For example: consider that 2 stations have packets ready to be transmitted - station1 with *data1* and station2 with *data2*, that will be transmitted at approximately the same time; a collision will occur and then both stations will backoff during a randomly selected delay between 0 and $2^n - 1$ slot times,

where n is the number of previous collisions. In the first collision resolution interval, if station1 waits 0 slot times and station2 waits 1 slot time, station1 will transmit *data1* and station2 will wait for the completion of such transmission. Supposing that station1 has other packets to be transferred, then, in the following collision the backoff time of station1 will be 0 or 1, and the backoff time of the station2 will be 0, 1, 2 or 3. Therefore, station1 will have a higher transmission probability. Such a *Packet Starvation Effect* will occur whenever a station has a sequence of packets to be consecutively transferred, if the network interface adapter is able to effectively contend for the network access at the end of every transmitted frame. Otherwise, one other station will acquire the transmission medium.

The major innovation of the BLAM protocol is the use of a modified collision counter policy. A BLAM station monitors the channel when it is in a backoff period, in order to use the knowledge of other station collisions and successes to modify its own backoff time. As a consequence, following a successful transmission, all the stations have an equal access probability to the medium. BLAM shows a large decrease in mean delay over BEB under moderate to high network load.

The Capture Avoidance Binary Exponential Backoff (CABEB) algorithm proposed by Ramakrishnan and Yang [59] also addresses the packet starvation effect. It enhances the collision resolution algorithm for the special case when a station attempts to capture the channel following an uninterrupted sequence of message transfers. Basically, the CABEB works as follows. When transmitting a second packet of an uninterrupted consecutive transmission, the station backoffs during 2 slot times after the first collision. If the other colliding station is transmitting a packet that has not experienced any collision, that station will select a backoff of 0 or 1 slot times and hence its packet is guaranteed to be transmitted. However, if other collision occurs after the backoff of 2 slot times (second collision), the CABEB station draws a backoff of 0 slot times. If the same packet experiences a third or subsequent collision, the CABEB station uses the standard BEB algorithm.

Finally, the traffic smoothing mechanisms proposed by Kweon *et al.* [60, 61] constraints the packet generation rate of non real-time messages below a defined threshold, in order to provide a probabilistic guarantee of message delivery. The traffic smoother is implemented between the IP layer and the Ethernet MAC layer (Figure 2.8).

The original traffic smoothing proposal uses the well-known leaky bucket

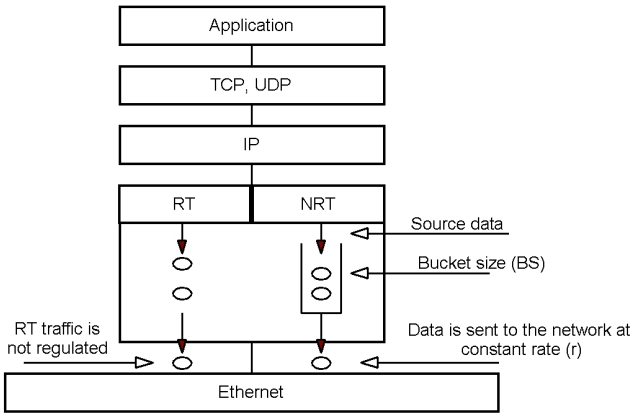

Figure 2.8: Example of Traffic smoothing.

regulator [62]. The leaky bucket has two parameters: the bucket size (BS) and the refresh period (RP). BS limits the maximum number of credits that can be stored in the credit bucket. If a packet arrives when the bucket is full, the packet is discarded. When a packet arrives and there is no credit in the bucket, the packet is kept in the queue until new credits arrive. The credits are added to the bucket every RP period. Packets are sent into the network at constant transmission rate (r), thus smoothing traffic bursts. On the other hand, it is assumed that the RT traffic is regulated at the system setup time. During the system run time, the RT traffic is no longer constrained.

Several policies for traffic smoothing have been proposed. The HIMD (Harmonic-Increase and Multiplicative Decrease) [61] is a dynamic policy that uses the credit bucket depth and the refresh period as a dynamic traffic regulator; in the absence of collisions, it periodically increases the input bound by periodically reducing the refresh period. In [63], the smoothing actions are performed by a fuzzy controller, where the network load is observed along determined time intervals, via the measurement of the throughput and of the number of occurring collisions.

2.3 RT Communication in IEEE 802.11 Wireless Networks

There is a strong similarity between wired IEEE 802.3 and wireless IEEE 802.11 networks, which is the use of the CSMA algorithm to manage the

medium access. When using the CSMA algorithm, each node verifies the absence of traffic before transmitting in the shared physical medium (electrical bus or band of electromagnetic spectrum). However, and contrarily to the case of wired networks, the CSMA Collision Detection procedure (CSMA/CD) cannot be used on wireless environments, as it would require the implementation of a full-duplex radio. As a consequence, the IEEE 802.11 standard [7] implements a Collision Avoidance procedure (CSMA/CA), that is referred as the *Distributed Coordination Function* (DCF).

In the following subsections, a number of relevant solutions to support real-time communication in IEEE 802.11 wireless networks are described, according to the same classification axis that was used for the case of wired IEEE 802.3 networks.

2.3.1 Avoiding collisions

The Point Coordination Function (PCF) is one of the main solutions intended to *avoid collisions* in IEEE 802.11 wireless networks. It has been proposed in the original IEEE 802.11 standard [7] as an optional access mechanism. It implements a centralized polling scheme to support synchronous data transmissions, where the Point Coordinator (PC) performs the role of polling master. Generally, the PC resides in the Access Point (AP). When the PCF scheme is used, the time scale is divided in two superframes consisting of a Contention Period (CP), used by the DCF scheme and a Contention Free Period (CFP), used by the PCF. Each CFP begins with a Beacon Frame sent by the PC. PCF has higher priority than DCF, since it may start the transmission after a shorter waiting time (PIFS). The PC takes control of the medium at the beginning of the CFP and maintains its control during the entire CFP. During the CFP period, the AP uses a polling scheduling algorithm to poll the stations. The polled stations respond to the polling packets; if the polled station does not have any pending packet, the response is a null frame with no payload.

The quest for real-time communication in IEEE 802.11 networks lead to the establishment of the IEEE 802.11e task group in July 1999. In December 2005, the Task Group E published the IEEE 802.11e amendment [8]. This amendment is an intended solution for real-time communication in wireless networks. It incorporates an additional coordination function called hybrid

coordination function (HCF), that is only used in QoS network configurations.

The HCF provides two mechanisms for the support of applications with QoS requirements: the *Enhanced Distributed Channel Access* (EDCA), which delivers traffic based on differentiating user priorities (UPs) and; the *Hybrid Coordination Function* (HCF) *Controlled Channel Access* (HCCA), which allows the reservation of transmission opportunities (TXOPs) with the hybrid coordinator (HC).

The HCCA mechanism was proposed to improve the PCF scheme. It is intended to guarantee bounded delay requirements, based on a Round Robin scheme. In contrast to the PCF scheme included in the legacy 802.11 MAC, the HCCA operates during both the CFP and CP periods (Figure 2.9).

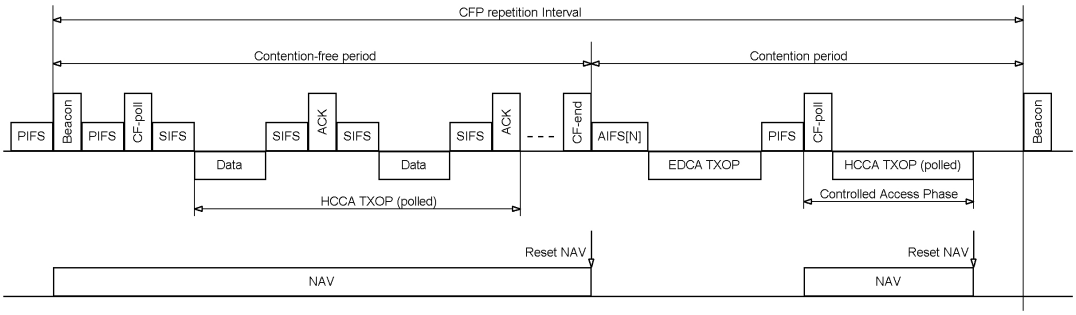

Figure 2.9: Example of CFP repetition interval.

The HC gains control of the wireless medium by waiting a shorter time between transmissions than the stations using the EDCA or DCF procedures. The HC may include a CF (Contention Free) parameter set element in the Beacon frame in order that all stations set their NAVs (Network Allocation Vectors) to the end of the controlled phase. During the CFP, the HC controls the access to the channel by polling all the stations in the polling list. To each polled station is granted a transmission opportunity (TXOP). On the other hand, the HC is also allowed to start a TXOP in the CP immediately after the channel is determined to be idle for one PIFS (Point Interframe Space) period, the called Controlled Access Phase (CAP). A CAP ends when the HC does not reclaim the channel after a duration of PIFS after the end of a TXOP.

Similarly to the PCF scheme, the HC also polls *all* the stations in the polling list, even though some stations may not have messages to transmit.

When the HC polls a station that has no packets to transfer, the station will transmit a null frame, after the QoS CF-poll. As a consequence, the polling overhead is roughly equal to the time interval from sending the polling frame till the end of the ACK frame [64]. Figure 2.10 shows the polling overhead in the 802.11e HCCA. On the left is represented the case where a station has data to transmit, and on the right is represented the case of a station without data to be transmitted.

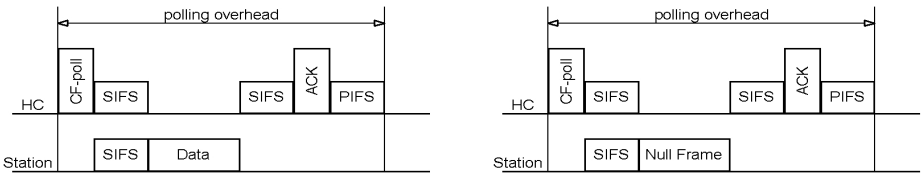

Figure 2.10: Polling overhead in the 802.11e HCCA.

A number of improvements have been proposed to reduce the HCCA polling overhead. For instance, Son *et al.* [64] proposed a polling scheme where the HC punish the stations that have no packets to transmit. When a station transmits a null frame, this station will not be polled again during a period of time.

Lo, Lee and Chen [65] designed a multipolling mechanism called *Contention Period Multipoll* (CP-Multipoll), which incorporates the DCF access scheme into the polling scheme. It uses different backoff values for the multiple message streams in the polling group, where each station executes the backoff procedure after receiving the CP-Multipoll frame. The contending order of these stations is the same as the ascending order of the assigned backoff values.

The first station in the polling list initializes its transmission immediately after receiving the CP-Multipoll frame. This action avoids the interference from other stations performing the backoff procedures in the DCF mode. Moreover, in order to avoid the repeated collisions between stations that are operating on the same channel in the overlapping space, the values assigned in the CP-Multipoll among neighboring BSSs must be different.

Lee *et al.* [66] proposed a polling scheme based on a master slave solution. This solution is implemented using a specific network driver (introduced by Microsoft) that has two basic functions: managing a network adapter, including sending and receiving data through the adapter; and interfacing

with higher-level drivers. The proposed solution uses an enhanced four-layer architecture that is organized into physical, data link, device driver, and application layer (Figure 2.11).

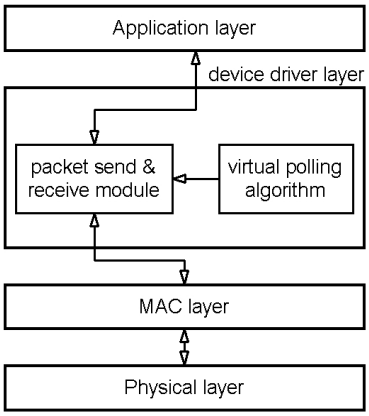

Figure 2.11: Four-layer architecture using NDIS.

A virtual polling list (VPL) contains the MAC address of the wireless slaves to be polled, and a virtual polling period (VPP) defines the duration of the polling cycle. When a slave receives a poll frame from the master, it can transmits a response frame to the master, or directly to another slave. Furthermore, after polling all the slaves registered in the VPL, the master invites other slaves into the network through the broadcast of an entry claim frame.

Other solutions based on token passing mechanisms have also been proposed. In [67], Ergen *et al.* proposed the WTRP (Wireless Token Ring Protocol). Basically, the WTRP is a MAC protocol that exchanges special tokens with additional fields and uses multiple timers to maintain the nodes synchronized. Besides, WTRP requires the joining node to be connected to its prospective predecessor and successor through a local connectivity table. The token is rotated around the ring, each station transmits during a specified time and if enough time is left, the station invites nodes outside the ring to join.

In [68], Willig presented the FTDMA (Flexible TDMA) MAC protocol. FTDMA is based on a polling scheme, where a base station polls all registered real-time stations in every frame. A frame is logically subdivided into phases: SYNC, Polling, Reservation, Register, Current Scheduler and Data Transfer. The main advantage of the FTDMA over traditional TDMA solutions is that unused slots can be used by other stations.

In [69, 70], Miorandi *et al.* proposed a solution based on a Master-Slave architecture on top of IEEE 802.11. In that proposal, cyclic packets are exchanged by means of periodic queries sent by the master to the slaves. Three different techniques were proposed to handle acyclic traffic: the first technique queries the slaves that signaled the presence of acyclic data, at the end of the current polling cycle. The second technique allows a slave, when polled, to send directly acyclic data to the master. The third one exploits the decentralized nature of the IEEE 802.11 MAC protocol. When acyclic data is generated, it allows a slave to immediately try to send data to the master.

Cheng *et al.* [71] presented a wireless token-passing protocol, named *Ripple*. Ripple uses six types of frames: DATA, NULL, RTS, CTS, ACK and Ready-To-Receive (RTR). The frame format of these frames are the same as those defined in 802.11 standard, except that Ripple only utilizes fixed-duration DATA frames. The RTR frame has the same format as CTS and it is used by a station to request a DATA frame from its upstream station. The IFS of RTR frame, is set as two SIFS plus the time needed to transmit an RTS frame. The IFSs of the remaining frames are all set as SIFS. Basically, Ripple modifies the data transmission procedure of 802.11 DCF and employs RTS and RTR frames as tokens. A station can only send a DATA frame if it holds a token.

2.3.2 Deterministic collision resolution

Concerning the *deterministic collision resolution* techniques, only those that are based on *forcing the resolution in favor of the RT station* have been adapted from IEEE 802.3 wired networks.

In [72], Sobrinho and Krishnakumar adapted the previously referred EQuB mechanism to wireless networks. The proposed *black-burst* (BB) mechanism is a distributed MAC scheme applied to *ad hoc* CSMA wireless networks (IEEE 802.11 DCF). It requires the shut-down of the random retransmission scheme. Real-time stations implementing the black burst approach contend for the channel access after a medium interframe spacing t_{med}, rather than after the long interframe spacing t_{long}, used by standard stations. Thus, real-time stations have priority over standard stations. When a real-time station wants to transmit, it sorts its access rights by jamming the channel with energy pulses (BB's) immediately after an idle

period of length t_{med}. The length of the BB transmitted by a real-time node is an increasing function of the contention delay experienced by the node.

A similar scheme is presented in [73], where voice nodes (real-time stations) use energy-burst (EB) (that is similar to BB) periods to prioritize real-time packets over data packets. The AP (Access Point) can transmit a VoIP packet after PIFS without backoff or contending. On the other hand, each voice station has its own address (ID), referred as VID (virtual identification). The VID can be assigned during the traffic stream (TS) setup procedure. The VID is expressed as binary value based on fixed total bits, which are determined by the voice packet resolution period (VPRP). The station with the highest VID wins the contention.

Figure 2.12 shows an example of the proposed scheme, where the VPRP is 9 slots including 3 EBs for bit separation, called as EBBS (energy-burst for bit separation). Therefore, when using 9 slots 64 (2^6) stations can be distinguished. The VPRP can be adjusted according to the number of real-time stations. The EBBS prevents new arrived VoIP packet from starting contention during VPRP, as two idle slots are not enough for $AIFS_v$. In the same way, data packets can not start the backoff procedure during VPRP.

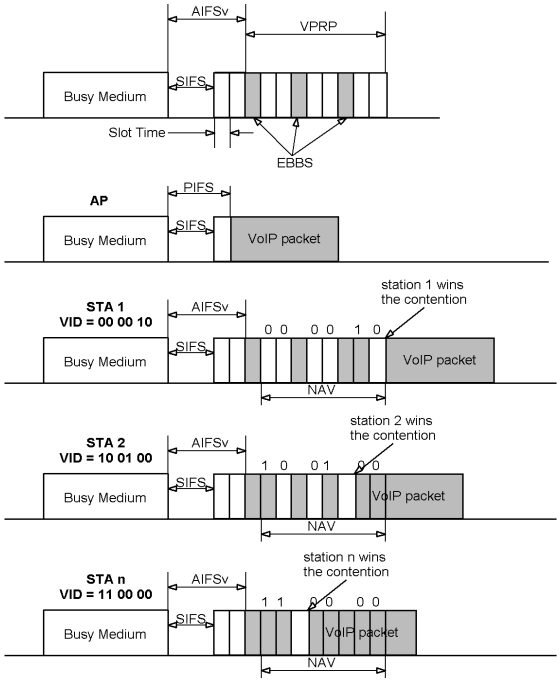

Figure 2.12: Energy burst approach.

Basically, when a station has a VoIP packet to transmit, it sends an EBBS after $AIFS_v$ and transmits EBs at position of 1 in the binary expression of its VID. The EBBS is sent if there are any $1s$ after that EBBS position. Each station in Figure 2.12 can transmit a VoIP packet at the designated time in case that there are no other stations having larger VID than itself. Station 1 has $VID = 2$ that is represented as 000010 in binary. Then, the station 1 transmits three EBBSs and one EB at the eighth slot correspondent to 1 position. One station wins the contention, whenever detects one idle slot after its last 1 position. Therefore, station 1 could start its VoIP packet only after the ninth slot. Similarly, station 2 transmits first EBBS and two EBs at 1 positions. Station 2 can send a VoIP packet after detecting no third EBBS, which means that there is no station having larger VID. Finally, according to the example, station n wins the channel in the fifth slot and transmits VoIP packet after detecting no second EBBS.

In [74], Shew *et al.* proposed a priority MAC protocol based on Sobrinho's approach complemented by a binary tree referred as *contention tree*. Basically, the black-burst (BB) scheme is adopted to distinguish the priorities of stations, and for stations with the same priority, they send messages in a round robin manner. The basic idea is that a station can obtain a unique ID number, which depends on its position in the contention tree.

2.3.3 Reducing the number of occurring collisions

The EDCA mechanism[2] available in IEEE 802.11 standard is specifically intended to reduce the number of occurring collisions. A possible solution to provide real-time communication under EDCA would be to use the highest access category (*voice*) to transfer real-time messages. However, using the EDCA mechanism to support real-time communications suffer from some severe limitations, specially when considering next generation communication environments characterized by an unknown number of communicating devices and an unpredictable network load. In a previous research work [22], we have assessed the behavior of this category when used to transfer small sized packets in an open communication environment. Both the number of packet losses and the average size of the MAC queues forecasted an unacceptable number of deadlines losses for real-time message streams, even for intermediate load cases. Such research work will be detailed in chapter 5.

[2] The EDCA mechanism will be detailed in chapter 3.

Hamidian and Körner [75] presented an interesting solution that provides QoS guarantees to the EDCA mechanism. The proposed solution, which is based on a draft version of the IEEE 802.11e [76], allows stations with higher priority traffic to reserve time for collision-free access to the medium. Basically, it proposes the transfer of the HCCA admission control and the scheduling algorithms from the HCCA controller to the contending stations.

The proposed approach uses the traffic specification (TSPEC) as defined in the draft version of the IEEE 802.11e HCCA standard [76]. The TSPEC is an element sent through a management frame that contains information about the characteristics and QoS expectation of a traffic stream. For instance, the maximum service interval specifies the maximum time interval between the start of two consecutive service periods. The scheduling and the admission control of a new traffic stream is locally made at each station, where it is calculated the scheduled service interval (SI) and the TXOP duration. Similarly to HCCA, it defines two contention periods (Figure 2.13). The first part is used as a contention-free period by high priority stations that have reserved TXOPs and the second part is used for low priority stations.

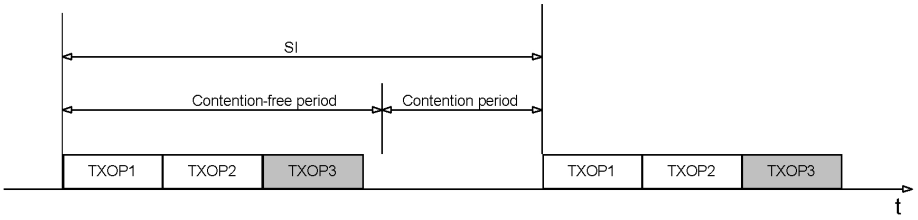

Figure 2.13: The scheduling of the reserved TXOPs.

Figure 2.13 shows an example where two stations have scheduled their TXOPs and a third station is about to schedule its own TXOP. When there are no TXOPs previously reserved, any high priority station can start a newly service interval. In such a case, the sender must broadcast the TSPEC element with all the information concerning the assigned values. Conversely, once SI and TXOP durations are calculated, the admission control is determined as follows. Considering k admitted streams, a new stream $(k+1)$ can be admitted if it satisfies the following inequality:

$$TXOP_{k+1} + \sum_{i=1}^{k} TXOP_i \leq SI - T_{CP} \tag{2.1}$$

where T_{CP} is the duration of the contention period. Unfortunately, it cannot coexist with any other PC or HC operating under the PCF or HCCA modes.

Wang *et al.* [77] designed a new collision resolution mechanism, referred as gentle DCF or GDCF. The difference between GDCF and DCF is that GDCF takes a more conservative measure by halving the CW (Contention Window) value only if there are c consecutive successful transmissions. Conversely, DCF reset its CW to the minimum value once there is a successful transmission. The GDCF needs to maintain a continuous successful transmission counter that is reset to zero after each collision. Then, when a collision occurs GDCF works similarly to DCF.

If there are c consecutive successful transmissions, GDCF will halve the CW and selects a backoff timer value uniformly from $[0, CW]$. If the channel is idle, GDCF also reduces the backoff timer by 1, the same as in DCF.

Yang and Vaidya [78] proposed the Busy Tone Priority Scheduling (BTPS) protocol. This scheme makes the following assumptions: (i) each station is capable to monitor the carrier status of the data channel; (ii) each station in idle state is capable to monitor two busy tone channels (BT1 and BT2) and lock onto the signal in the data channel as desired; (iii) it is assumed that the two busy tone signals (BT1 or BT2) sent by a station can be sensed by other stations within the interference range of the former station. In BTPS, busy tone serves as the indication of backlogged high priority packets.

BTPS works similarly to the IEEE 802.11 DCF, with the difference that high priority and low priority behave differently during *IFS* and *backoff* stages. The BTPS protocol uses DIFS as the IFS for high priority stations. However, during DIFS and backoff stages, the high priority stations with queued packets send a BT1 pulse every M slots, where M is a constant. Between two consecutive busy tone pulse transmissions, there should be at least one empty SlotTime interval as the station must have a chance to listen to the data channel. Therefore, M could be any value larger than or equal to 2 and, the IFS of low priority stations should be larger than M slots, in order to enable sensing the busy tone signal. Figure 2.14 shows an example of the BTPS protocol, where the high priority stations send BT1 every M (M = 3 in the example) slots during DIFS and backoff stages.

In the example, it is considered that station 1 is a high priority station with queued packets. Then, all stations that senses a BT1, except those backlogged high priority ones, will send a BT2 pulse if it is not receiving

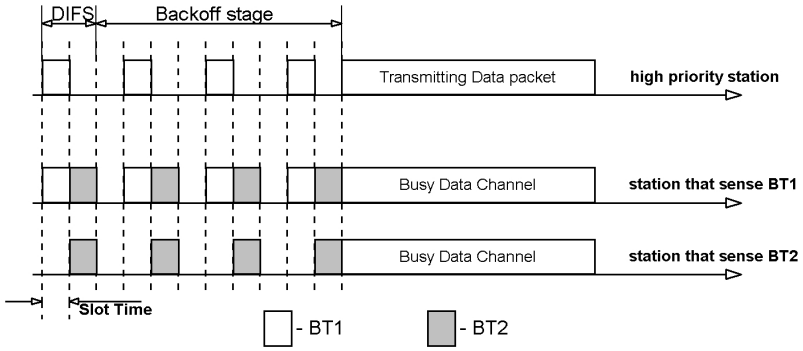

Figure 2.14: BTPS protocol.

a packet from the data channel. It will also defer its transmission of low priority packets. Afterwards, any station that sense a BT2 will also defer its transmission attempt of low priority packet for a RTS/CTS handshake duration. As a consequence, the IFS for low priority stations must be set to $(M + 1) \times SlotTime + SIFS$ to avoid priority inversion.

In [79], it was proposed a distributed algorithm intended to provide fair scheduling in a wireless LAN, referred as DFS (Distributed Fair Scheduling). The DFS protocol behaves quite similar to IEEE 802.11 DCF, except in what concerns the backoff interval that is initially calculated. The fundamental difference is that each station maintains a local virtual clock and, the backoff interval is chosen proportionally to the finish tag of the packet to be transmitted. The finish tag is calculated similarly to the SCFQ (Self-Clocked Fair Queueing) algorithm [80].

In [81], Lopez-Aguilera *et al.* evaluated the performance of the IEEE 802.11e EDCA when its working procedure is unsynchronized. The authors proposed the use of AIFS times values that are separated by values that are not multiple of the slot time. As a consequence, it would become possible to avoid collisions between frames from different access categories.

Lo Bello *et al.* [82] proposed a wireless traffic smoother (WTS) to support soft real-time traffic over IEEE 802.11 WLANS. The presented solution is similar to the traffic smoother scheme previously proposed for Ethernet networks [63].

Finally, the underlying traffic separation mechanism that is proposed in this thesis (chapter 3) has a similar behavior to both the SVP protocol [83] and the mechanism proposed by Hwang and Cho [84]. The SVP protocol specifies a backoff value of zero for stations or classes with the highest

priority level. A shortcoming is that if multiple SVP stations attempt to transmit at the same time, consecutive collisions will occur and a failure will be reported. The mechanism proposed in [84] consists in allowing the transmission of voice packets (highest priority) in the first empty slot in the first retransmission. When the first retransmission fails, the second retransmission performs the original backoff procedure. Therefore, these two approaches are not able by itself to provide real-time guarantees to the supported applications.

2.4 Synthesis of the State-of-the-Art

In the previous subsections, some of the most relevant CSMA-based real-time communication proposals have been classified according to a first classification axis. Such first axis is related to *how collision are dealt with*. In this subsection, those RT communication proposals are analyzed according to a second classification axis, that is related to the achieved *compatibility degree*. Specifically, this second axis highlights how the proposed RT communication solutions keep or alter the compatibility with IEEE 802.3/802.11 compliant devices.

As explained before, three different compatibility levels have been defined: *level 1*, *level 2* and *level 3*. The main characteristic of compatibility *level 1* is the impossibility of coexistence between enhanced (real-time) and default devices in the same network domain, unless that *all* communicating devices (both real-time and non real-time) are implementing the same enhancements. On the other hand, *level 2* and *level 3* subclasses comprise RT communication proposals able to offer RT guarantees in presence of *third* devices. The main difference between these two subclasses is related to the required level of modifications. The implementation of a level 2 device requires the use of specific hardware, impairing the use of COTS (commercial off-the-shelf) hardware. Conversely, a level 3 device can be implemented upon COTS hardware, requiring just modifications at the firmware/software level of the *real-time* communicating devices.

Figures 2.15 and 2.16 classify the reported RT communication proposals according these two axes, for wired IEEE 802.3 and wireless IEEE 802.11 networks, respectively.

Most part of the RT solutions that follow the *avoiding collisions* strategy are based on TDMA, Token-Passing, Master-Slave or Polling techniques. A

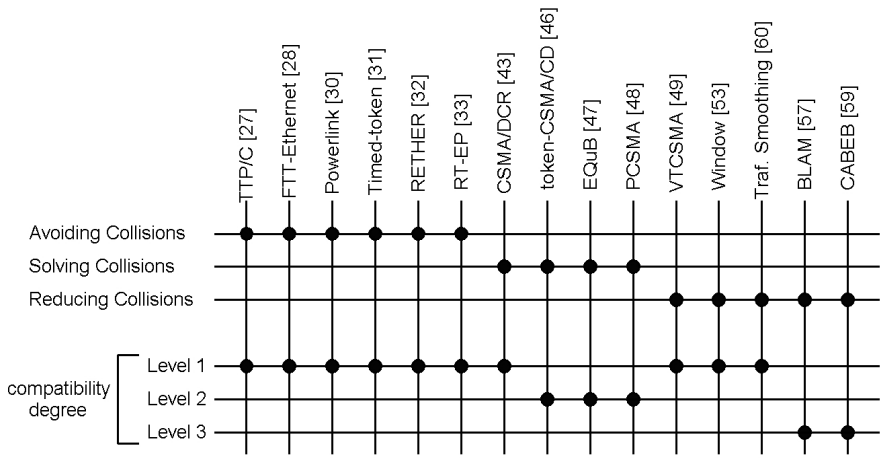

Figure 2.15: Supporting RT communication in IEEE 802.3.

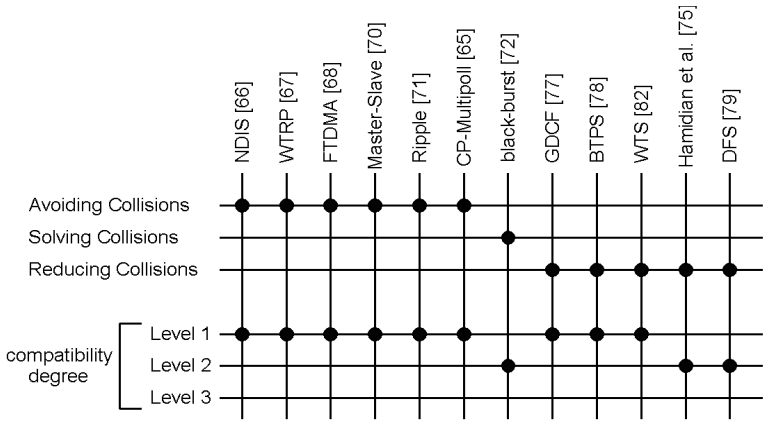

Figure 2.16: Supporting RT communication in IEEE 802.11.

common characteristic of most part of these RT solutions is that an enhanced (real-time) station is not able to support RT communication in the presence of standard IEEE 802.3/802.11 stations (unless these standard stations do not initiate any communication). That is, the majority of the avoiding collision solutions have a compatibility degree *level 1*, which impairs its use whenever *third* stations try to access the shared communication medium. Relevant exceptions are: the Switched Ethernet approach and the improvements included in the HCF (PCF and HCCA) mechanism of the IEEE 802.11e amendment.

However, the Switched Ethernet violates the decentralized paradigm

of the CSMA protocol. The Switched Ethernet enables the micro-segmentation of the network, thus each station operates in a Carrier Sense *Single* Access method. Therefore, the Switched Ethernet is out of the scope of this survey. On the other hand, despite PCF being well suited to handle delay-sensitive applications, most part of the WLAN network cards never actually implemented the PCF scheme, due to complexity reasons [69]. Therefore, the PCF mechanism has not been a solution to support RT communication, due to the unavailability of WLAN network cards. The HCCA mechanism [8] has been proposed as an improvement to the PCF mechanism. However, preliminary studies [85, 86] have already shown that the HCCA mechanism may not be suitable to guarantee the special requirements of industrial real-time applications. Furthermore, it is still not clear if the HCCA mechanism will be implemented in next generation WLAN network cards, solving the unavailability problem of the PCF mechanism.

Besides these two approaches, all other avoiding collision approaches require the strict control of the communication environment. Otherwise, they are not able to work properly, as they are not able to handle messages sent by *third* stations. There are also some additional reasons that impose the need for a strict control of the communication environment. For instance, TDMA-based solutions require a precise clock synchronization, in order to precisely define the communication slots. Master-Slave solutions rely upon the correct behavior of the master, which represents a single point of failure for the communication architecture. The Token-based approaches rely upon the correct token behavior. Similarly, in the Polling based schemes, all frames are required to pass through a *central coordinator*, wasting bandwidth and making the communication inefficient, which is also a single point of failure for the RT architecture.

Concerning the second sub-class (*solving collisions*) that enables the support of real-time communications in shared broadcast environments, its techniques are based on: i) *modifying* the MAC layer to achieve upper-bounded access time to the shared communication environment, or ii) *forcing* the collision resolution in favor of the RT stations that are trying to initiate a message transfer. Whatever the approach all those solutions that cannot be implemented in COTS hardware can be technically very interesting but are economically not viable today [5]. That is the main disadvantage of the CSMA/DCR protocol [43], for example. Besides that, the CSMA/DCR protocol has compatibility degree *level 1*, which impairs its use whenever the set of communicating devices is not previously fixed.

Summing up, enabling the *coexistence* between RT stations and *third* stations (that are out of the sphere-of-control of the RT architecture) is of utmost importance when dealing with the next generation of communication environments. Therefore, all approaches with compatibility degree *level 1* are not adequate. For instance, using *level 1* approaches in a wireless medium will hardly work, due to the open characteristics of the wireless medium. It is worth mentioning that, any wireless communication environment is susceptible to interferences created by other systems, not only from those using the same technology, but also from other technologies working in the same frequency band [11]. As a consequence, the ability to support next generation communication environments will strongly rely upon technical solutions with compatibility degree *level 2*, being desirable that solutions *level 3* arise.

Relevant exceptions based upon *solving collision* techniques with compatibility *level 2* are those solutions based on the black-burst (BB) scheme. This kind of solution consists in *forcing* the collision resolution in favor of the RT stations. The *forcing approach* is the most promising solution to provide RT communication in the next generation communication environments. Specifically, in communication environments characterized by a communication medium shared with timing unconstrained devices that generates an unpredictable network load. To out best knowledge, the forcing approach is the only technique that allows the coexistence of RT stations with an unknown number of communicating devices and an unpredictable network load, being able to prioritize RT communication in such hostile environments.

Analyzing the specific *forcing-collision* based approaches presented in this chapter, the main drawback of the hybrid token-CSMA/CD mechanism is that it requires the stations to be synchronized. Furthermore, it is also not compatible with COTS Ethernet hardware. Conversely, the EQuB is a very interesting technique that enables a privileged access to real-time traffic with a FCFS (First-Come-First-Serve) discipline. The collision resolution mechanism for real-time devices requires the disabling of the exponential backoff mechanism and the transmission of jamming sequence (BB) with durations dependent on the contention periods experienced by the real-time traffic. Another interesting approach is the PCSMA protocol, which adds extra bits in the real-time packet. When a collision with non real-time packet occurs, such extra bits guarantee that it has not been transmitted useful data before the collision resolution. For the case of wireless networks,

there are some adaptations of the black-burst scheme for forcing the collision resolution in favor of RT station. Basically, all solutions uses a jamming signal to prioritize real-time packet over data packets. These are, in our opinion, the most interesting approaches to support RT communication, as they are able to handle the requirements imposed by the next generation communication environments.

Finally, concerning the last subclass (*reducing collisions*), it is worthwhile to mention the solutions that constrain the generated traffic in a fair way, without making any further modification in the MAC protocol. The solutions like [49, 50, 53, 56, 57, 59, 60, 61] increase the network access fairness and reduce the collision number based on some priority criterion. The main drawback of all these solutions is the compatibility degree *level 1*. For instance, the traffic smoothing approach requires the smoothing strategy to be implemented in all the communicating devices. The BLAM and CABEB protocols can be highlighted in this subclass of protocols. Both protocols show a large decrease in mean delay over the traditional CSMA/CD protocol, and they have compatibility *level 3*. Unfortunately, they cannot provide real-time guarantees to the supported traffic.

2.5 Summary

Ethernet and WiFi are well-known and extensively used network technologies for LANs today. However, one of their main disadvantages is the inherent non-determinism of its probabilistic contention resolution. Such a probabilistic behavior impairs both communication standards to provide a real-time service to the supported applications, unless additional functionalities are introduced.

On the one hand, there are several techniques to support real-time (RT) communications for Ethernet networks. However, few of those techniques allow standard devices to coexist with enhanced (real-time) stations in the same communication environment (compatibility levels 2 and 3). Thus, all the network stations must be under the strict control of the real-time communication architecture (compatibility level 1). Otherwise, the RT-communication might be disturbed. Relevant exception is the Switched Ethernet approach. However, this solution have some drawbacks that were described before (buffer overflow, lack of adequate number of priority levels, etc). Furthermore, the communication is no longer based on the CSMA pro-

tocol, but on the use of a centralized communication scheduler: the Switch. As a consequence, the interest of shared Ethernet is still alive, mainly for applications requiring frequent multicast or for applications requiring precise control of transmission timings [40].

On the other hand, there is a trend for the implementation of RT communication systems on top of wireless networks, and, specifically, on top of WLAN [9, 87]. A fundamental assumption that must be considered is that the wireless communication medium is a relatively *open communication environment*, *i.e.*, any new participant can try to access the communication medium at any instant (according to the MAC rules) and establish its own communication channels. Therefore, the traditional approaches that guarantees RT behavior through the tight control of every communicating device (compatibility level 1) are no longer applicable. For instance, the use of traffic smoothers [61, 63] is not adequate for wireless environments, since it is not possible to impose any traffic smoothing strategy upon stations that are out of the sphere-of-control of the RT architecture. Besides, most of the RT communication approaches proposed for both Ethernet and WiFi networks cannot be implemented using COTS hardware (compatibility level 2). Such communication approaches can be technically very interesting, but are economically not viable today [5].

Summing up, the coexistence of standard devices together with enhanced (real-time) devices is a hard task. Specifically for the case of wireless networks, the underlying wireless communication protocol must be able to guarantee the timing constraints of the RT traffic in a communication environment shared with timing unconstrained devices that generate an unpredictable traffic load. Therefore, the most promising solutions to provide RT communication in the next generation of wireless networks are those that force the collision resolution in favor of the RT station (*e.g.* the black-burst technique [72]).

Within this context, a new RT-communication approach based on forcing collision resolution technique is proposed in this thesis. The proposed approach is based on traffic separation mechanisms. Such mechanisms are able to prioritize RT-traffic over timing unconstrained traffic, without directly controlling the latter. That is, instead of controlling *all* the traffic generated by *all* the stations, the proposed approach will only control the traffic generated by the RT stations, in such a way that RT-traffic will have priority over other traffics. Besides, no hardware/software changes are necessary to stations that are out of the sphere-of-control of the RT com-

munication architecture. This requirement is imposed by the large installed base of standard IEEE 802.3/802.11 devices and from the fact that wireless media is an open communication medium. Therefore, it would not be realistic to assume that is possible to create a zone free of standard IEEE 802.11 stations.

Chapter 3

A New Traffic Separation Mechanism (TSm)

In this chapter, a new Traffic Separation mechanism (TSm) for CSMA networks is presented. The TSm mechanism allows the coexistence of IEEE 802.3 or IEEE 802.11e standard stations with enhanced (real-time) stations in the same network domain (wired or wireless environments), prioritizing the real-time traffic. This mechanism enables the implementation of an inexpensive "forcing collision resolution" scheme, that solves the collisions in favor of RT stations using COTS hardware (compatibility level 3). This chapter is largely drawn from the following published work: "A Probabilistic Analysis of Traffic Separation in Shared Ethernet Systems Using the h-BEB Collision Resolution Algorithm" (Moraes and Vasques [17]); "Real-Time Traffic Separation in Shared Ethernet Networks: Simulation Analysis of the h-BEB Collision Resolution Algorithm" (Moraes and Vasques [88]); "Probabilistic Timing Analysis of the h-BEB Collision Resolution Algorithm" (Moraes and Vasques [16]); "A Traffic Separation Mechanism (TSm) allowing the coexistence of CSMA and real-time traffic in wireless 802.11e Networks" (Moraes et al. [15]); "A forcing collision resolution approach able to prioritize real-time traffic in CSMA-based networks" (Moraes et al. [14]).

3.1 Introduction

As discussed in chapters 1 and 2, the most promising solutions to provide RT communication in the next generation communication environments are those that force the collision resolution in favor of RT stations. The TSm mechanism has been proposed within this context. It is able to prioritize real-time traffic over timing unconstrained traffic, without directly controlling the latter. That is, instead of controlling *all* the traffic generated by *all* the stations, the proposed mechanism will control only the traffic generated by the stations supporting RT-traffic, prioritizing this traffic over the timing unconstrained multipurpose traffic. Furthermore, the proposed TSm mechanism is one of the most simple schemes with compatibility *level 3*, providing an inexpensive solution to be used in shared broadcast environments.

The TSm mechanism is not intended to be used just by itself to support real-time communication. The main reason is that this mechanism, as it is, is able to prioritize the real-time traffic of just one station in the network. Examples of RT communication architectures built upon the TSm mechanism are the VTPE-hBEB (proposed in chapter 4) and the VTP-CSMA (proposed in chapter 6) architectures. These architectures are able to support RT communication among multiple stations in Ethernet (IEEE 802.3) and WiFi (IEEE 802.11) networks, respectively. Concisely, the proposed TSm mechanism follows the trend identified in the state-of-the-art review (chapter 2).

The remainder of this chapter is organized as follows. Section 3.2 describes how the CSMA protocol work, for the case of collision detection (CSMA/CD) and collision avoidance (CSMA/CA). Section 3.3 highlights the proposed Traffic Separation mechanism (TSm), focusing on the implementation details that must be built upon the IEEE 802.11e and 802.3 standard devices. Then, the simulation and analytical studies are discussed, followed by some conclusions.

3.2 The CSMA protocol

The protocol used to control the bus access is the key issue in any broadcast network. The Carrier Sense Multiple Access (CSMA) protocol defines a medium access control family of protocols, where stations contending for

the access to a shared medium must listen before transmitting. Basically, this family of protocols has the following behavior:

- When a station wants to transmit, it listens to the transmission medium;
- If the medium is idle, the station will start the transmission (either immediately, or after a defined interval, depending on the specific protocol);
- If the medium is busy (*i.e.* another station is transmitting), the station will defer its transmission to a later time instant that depends on the protocol being used;
- A collision will occur whenever two (or more) stations sensed the medium free and decided to simultaneously transmit.

The CSMA medium access methods that are implemented by different communication protocols differ on how the waiting time intervals before transmitting are evaluated, either after sensing the medium idle, or before re-transmitting after a collision.

3.2.1 The CSMA/CD protocol

The CSMA with *Collision Detection* (CSMA/CD) is the protocol implemented at the MAC layer of both IEEE 802.3 [24] and Ethernet local area networks. At this layer, frames are transferred by the IEEE 802.3 or the Ethernet standards with, respectively, the following formats (Figures 3.1 and 3.2):

Figure 3.1: The 802.3 frame format.

IEEE 802.3 and Ethernet are almost similar. The main difference is the logical control sublayer, which is absent in IEEE 802.3 (it is covered by IEEE 802.2 [89]). However, both standards are compatible, and Ethernet is used as a popular name for IEEE 802.3. In this thesis, the terms

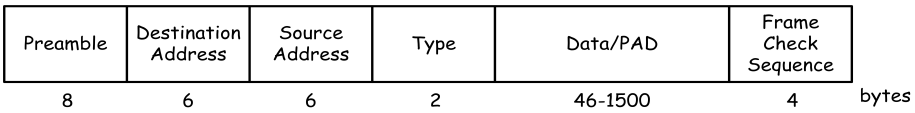

Figure 3.2: The Ethernet frame format.

IEEE 802.3/Ethernet are used interchangeable to designate the IEEE 802.3 standard. For a 10/100 Mbps IEEE 802.3 implementation, the used set of parameters is presented in Table 3.1.

Table 3.1: IEEE 802.3 parameters.

Parameters	**Values**	**Bit Rate**	
		10 Mbps	100 Mbps
SlotTime	512 bits	51.2 μs	5.12 μs
InterFrameGap	96 bits	9.6 μs	0.96 μs
JamSize	32 bits	3.2 μs	0.32 μs
AttemptLimit	16	-	-
BackoffLimit	10	-	-
MaxFrameSize	12208 bits	-	-
MinFrameSize	512 bits	-	-
AddressSize	48 bits	-	-

Basically, the CSMA/CD protocol implemented by both IEEE 802.3 and Ethernet standard protocols works as follows: whenever a station has a message to transfer, if the transmission medium is idle, it will immediately transmit. If a collision is detected, all the transmitting stations terminate their transmissions and send a jamming sequence (to ensure that all the transmitting stations recognize the collision[1]). When the transmission is aborted due to a collision, its retransmission will be retried after a randomly evaluated delay (backoff time) until it is, either successfully transmitted, or eventually discarded (after a maximum number of 16 attempts).

One of the key issues of the CSMA/CD protocol is the evaluation of such backoff delays, which is locally done by executing the *Binary Exponential Backoff* (BEB) algorithm. This algorithm operates as follows: after the end of the jamming sequence, the time is divided into discrete slots, whose length is equal to the slot time ($SlotTime$). The backoff time is given by

[1]More accurately, when detecting a collision, the station always finishes the transmission of the Preamble and the Start of Frame Delimiter (64 bits), if these have still not been completely transmitted. Afterwards, it transmits a jamming sequence (32 bits), and then stops.

$t_{backoff} = r \times T$, where r is a random integer in the range $0 \leq r \leq 2^k - 1$, k is the smaller of n or 10 and T is the slot time in seconds. This means that the station will wait between 0 and $2^n - 1$ slot times before retransmitting, being n the number of collision resolution rounds. Finally, after 10 attempts, the maximum waiting interval is fixed at 1023 slot times, and after 16 attempts a failure is reported and the transmission is aborted (Figure 3.3).

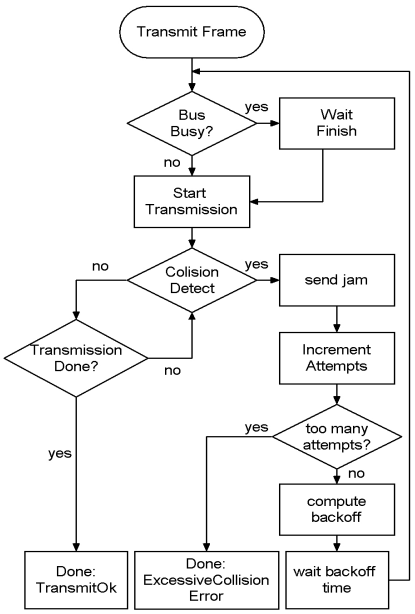

Figure 3.3: Control Flow Summary - CSMA/CD.

The CSMA/CD protocol seems to have a random queue service discipline, *i.e.*, the message to be transferred after a successful transmission seems to be randomly chosen among the total number of hosts with ready messages. However, Christensen [90] demonstrated that the BEB algorithm imposes a last come first serve policy, as a station with the more recently queued packet, will have a higher probability for the acquisition of the medium. Another particularity of the CSMA/CD protocol is the *Packet Starvation Effect* that was previously explained in chapter 2.

3.2.2 The CSMA/CA protocol

The Collision Detection procedure cannot be used in wireless LAN networks, as it would require the implementation of a full-duplex radio, capable of simultaneously transmitting and receiving. As a consequence, the medium

access mechanism of the IEEE 802.11 standard [7] is the CSMA with *Collision Avoidance* (CSMA/CA), also called *Distributed Coordination Function* (DCF).

More accurately, the IEEE 802.11 MAC sublayer introduces two medium access coordination functions, the mandatory DCF and the optional Point Coordination Function (PCF). DCF[2] is the basic mechanism of the IEEE 802.11. When a station wants to transmit, the station senses the medium (carrier sensing), if the medium is idle during a specific time interval (called DIFS, Distributed Interframe Space), it immediately transmits, and other stations wait until medium becomes idle again at least for DIFS time period. If the medium is busy, the stations selects a random number, in the range of $[0, CW]$, where CW is initially assigned as CW_{min}. The CW parameter will be increased whenever a transmission fails, *i.e.*, the destination station does not respond with an acknowledgement (ACK frame). After an unsuccessful transmission attempt, another backoff interval will be selected, where the CW value is increased by $[(oldCW + 1) * 2 - 1]$, with an upper bound given by CW_{max}. On the other hand, the backoff timer decrements the backoff interval each time the medium is detected to be idle. As soon as the backoff timer becomes zero, the station can retry its transmission (Figure 3.4).

In addition to the DCF mechanism, the IEEE 802.11 MAC sublayer also defined the Point Coordination Function (PCF), which uses a centralized polling scheme to support synchronous data transmissions upon the DCF mechanism. The PCF scheme was previously described in chapter 2.

Recently, the IEEE 802.11e standard was published [8] as an amendment to the original standard, intended to provide Quality of Service (QoS). This amendment incorporates an additional coordination function called Hybrid Coordination Function (HCF) that is only used in QoS network configurations (Figure 3.5). The HCF mechanism schedules the access to the channel by allocating transmission opportunities (TXOP) to each of the stations. Each TXOP is defined by a starting time and a maximum length and may be obtained through one of two access mechanisms specified by the HCF: the *Enhanced Distributed Channel Access* (EDCA) and the HCF *Controlled Channel Access* (HCCA) [8].

The underlying idea of the EDCA mechanism standard was firstly proposed by Deng and Chang [91], where the higher priority class uses the win-

[2] An additional mechanism, RTS/CTS, is defined in the IEEE 802.11 standard to solve the hidden terminal problem and to obtain a better behavior for transmission of long message. For further details, please refer to [7].

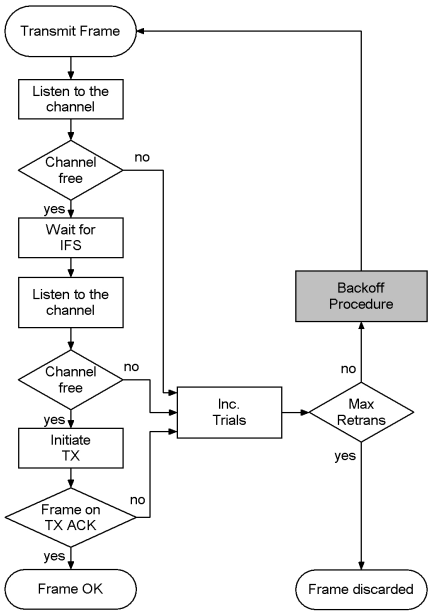

Figure 3.4: Control Flow Summary - CSMA/CA.

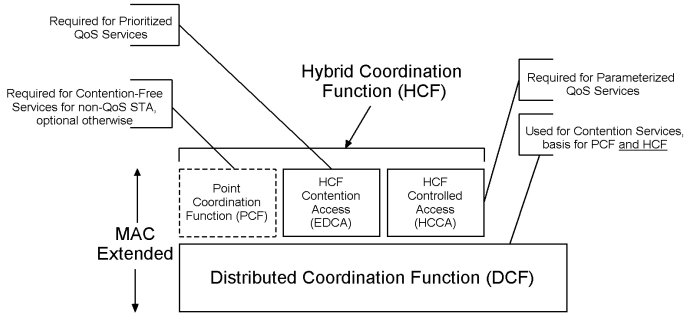

Figure 3.5: IEEE 802.11e MAC architecture.

dow $[0, 2^{j+1}-1]$ and the lower priority class uses the window $[2^{j+1}, 2^{j+2}-1]$, where j is the backoff stage.

The EDCA function implements a CSMA/CA mechanism for the channel access under the control of the HCF coordination function. It is designed to provide differentiated transmission services, with 4 priority levels. It enhances the DCF scheme, as each frame arriving at the MAC layer with a defined priority will be mapped into one of the four access categories (AC). These ACs are based on the 8 priority levels defined by the IEEE 802.1D standard.

Different levels of service are provided to each of the AC traffics, based

on three independent mechanisms: (i) the Arbitration Interframe Space (AIFS); (ii) the TXOP time interval and; (iii) the Contention Window size (CW). The default values used in the EDCA mode for $AIFNS$, CW_{min} and CW_{max} are presented in Table 3.2, where the aCW_{min} and aCW_{max} parameters are usually set to, respectively, 31 and 1023 slot times.

Table 3.2: Default EDCA parameter set.

AC	CW_{min}	CW_{max}	**AIFSN**
AC_VO	$(aCW_{min}+1)/4-1$	$(aCW_{min}+1)/2-1$	2
AC_VI	$(aCW_{min}+1)/2-1$	aCW_{min}	2
AC_BE	aCW_{min}	aCW_{max}	3
AC_BK	aCW_{min}	aCW_{max}	7

Firstly, for a station operating under EDCA, each frame will wait during an *AIFS[AC]* interval, instead of waiting during a DIFS interval (as it was the case for DCF in IEEE 802.11). Only after the channel remaining idle during an *AIFS[AC]* interval, the station will start to transmit the frame. The duration of the *AIFS[AC]* interval is given by:

$$AIFS[AC] = AIFSN[AC] \times aSlotTime + aSIFSTime \tag{3.1}$$

where the *AIFSN[AC]* must be greater than or equal to 2 for all stations, except for the QoS Access Points (QAPs) where it shall be greater than or equal to 1. The values of $aSlotTime$ and $aSIFSTime$ depends on the physical characteristics of the channel. For example, for the IEEE 802.11a PHY mode the shortest interframe space (SIFS), which is defined by the variable $aSIFSTime$, is equal to $16\mu s$ and, $aSlotTime$ is equal to $9\mu s$. Figure 3.6 shows the relationships between the multiple *AIFSs* in the EDCA scheme.

Additionally, the EDCA mechanism introduces the TXOP concept, *i.e.*, a time interval during which the station keeps the medium access control (Table 3.3). Consequently, multiple frames may be transmitted within an acquired TXOP, if there is more than one frame pending to be transferred in the AC for which the channel has been acquired.

Finally, if a station wants to transmit a frame while the channel is busy, or becomes busy before the expiration of the *AIFS[AC]*, the backoff procedure is invoked (third traffic differentiation mechanism). The contention

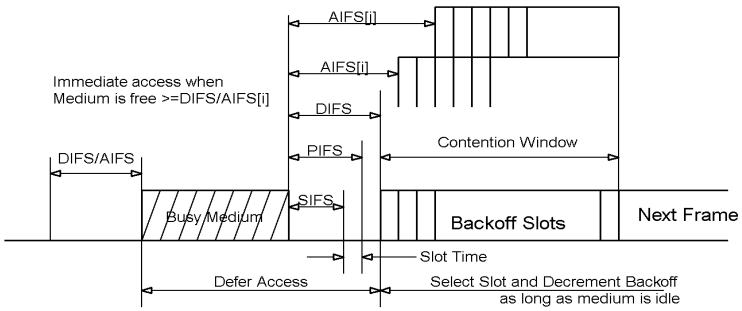

Figure 3.6: Interframe spaces in the EDCA mechanism.

Table 3.3: TXOP default EDCA values.

AC	**TXOP 802.11b**	**TXOP 802.11 a/g**
AC_VO	$3.264ms$	$1.504ms$
AC_VI	$6.016ms$	$3.008ms$
AC_BE	$0ms$	$0ms$
AC_BK	$0ms$	$0ms$

window is defined by the $aCW_{min}[AC]$ and $aCW_{max}[AC]$ attributes, in contrast to the legacy DCF where the initial values are randomly selected among the $[0, CW]$ interval defined by the physical layer. In the EDCA mechanism, the backoff procedure selects a random number, in the range *[0, CW]*, where the CW size is initialized at $aCW_{min}[AC]$. When a transmission fails, CW is increased by *[(oldCW[AC]+1)*PF] - 1* upper bounded by $aCW_{max}[AC]$, where PF is the persistence factor (its default value is *PF=2*). On the other hand, the backoff counter decreases the backoff interval whenever the medium is detected to be idle for *AIFS[AC]*. In contrast to the DCF mode, where the station would try to transmit as soon as the backoff timer reaches zero, in EDCA the station initiate the transmission in the first slot boundary after the backoff counter has reached zero. Moreover, DCF counters decrement at the end of idle backoff slots, while EDCA backoff counters decrement on *slot boundaries*. Consequently, there is no difference on the initial transmission time between DCF and EDCA, considering the same number of selected slots to backoff. Figure 3.7 illustrates the difference between the two decrementing procedures (DCF and EDCA).

The HCCA mechanism and its limitations was also previously described in chapter 2.

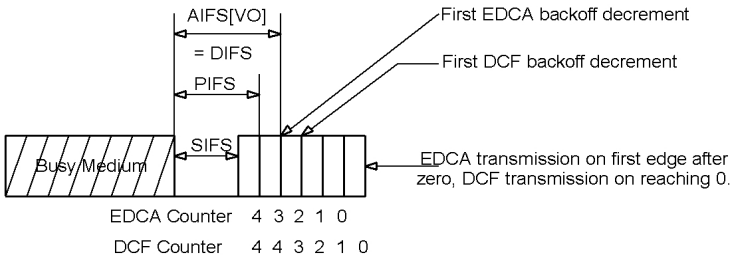

Figure 3.7: Decrementing procedures in DCF and EDCA.

3.3 The Traffic Separation mechanism (TSm)

3.3.1 Rationale

The target of this chapter is to propose an adequate traffic separation mechanism to handle RT communication in IEEE 802.3/802.11 communication networks. The proposed solution consider a heterogeneous environment consisting of stations with multipurpose communication entities (ST stations), *i.e.* standard IEEE 802.3 or IEEE 802.11 stations, and stations with real-time communication entities (RT stations), that share the same communication domain (wired or wireless environments). The network load imposed by the set of ST stations is out of the sphere-of-control of the RT communication architecture. Besides that, no hardware/software changes should be necessary to ST stations and, the proposed solution might be implemented in COTS hardware. These requirements arise from the large installed base of standard IEEE 802.3 and 802.11 devices, and also from the fact that wireless media is a relatively open communication medium. It is not realistic to assume that is possible to create a zone free of standard IEEE 802.11 stations in any wireless environment.

To address this problem, we propose a new real-time communication approach based on traffic separation mechanisms. Such mechanisms are able to prioritize RT-traffic over ST-traffic, without directly controlling the latter. That is, instead of controlling *all* the traffic generated by *all* the stations, a *level 1* approach, the proposed approach will control only the traffic generated by the RT stations. That is, it prioritizes the RT-traffic over the other traffic. Moreover, it can be easily implemented using COTS hardware, a *level 3* approach.

3.3.2 Implementation of the TSm mechanism

When considering the traditional CSMA/CD/CA protocols, whenever there is a collision, the collision resolution algorithm delays the retransmission during a time interval referred to as backoff delay. Such a backoff delay is a probabilistic function of the number of previous collisions, which means that the retransmission probability does not depend on the type of traffic, but just on the state of the collision counter of each particular station. Conversely, the proposed traffic separation mechanism ensures that whenever a collision occurs, either the RT message is transferred before any other message, or none of the messages is transferred at all.

Firstly, consider the TSm scheme applied to IEEE 802.3 networks (shared Ethernet). The TSm scheme is also hereafter referred as "high priority Binary Exponential Backoff (h-BEB)". A RT station implementing the h-BEB collision resolution algorithm (Figure 3.8) has the same operating behavior of a CSMA/CD station, except in what concerns the evaluation of the backoff delay. Whenever there is a collision, the RT station starts immediately the transmission (backoff interval equal to zero) after the end of the jamming sequence. This behavior guarantees the highest transmitting probability to the h-BEB station in a shared Ethernet segment. The h-BEB station will always try to transmit its frame in the first available slot, while all the other IEEE 802.3 stations that implement the BEB algorithm will wait during 0 and 2^n-1 slot times, where n is the number of collision resolution rounds.

Therefore, the h-BEB collision resolution algorithm is able to impose real-time traffic separation, as the traffic generated by the h-BEB station will always be transferred before the traffic generated by other stations. Basically, the modification for the h-BEB implementation consists just in setting the backoff delay parameter to 0 in the h-BEB station.

Secondly, consider the TSm scheme applied to IEEE 802.11 networks. Figures 3.9 (a), (b) and (c) summarize the dynamic behavior of the CSMA/CA protocol working, respectively, with DCF, EDCA and TSm modes.

As mentioned before the DCF mode, represented in Figure 3.9 (a) does not provide any traffic differentiation. Whenever the network is working in DCF mode and a collision occurs, all the involved stations will select a locally computed random backoff interval; subsequently, the stations may only retry to transmit whenever the backoff counter reaches 0. In the EDCA

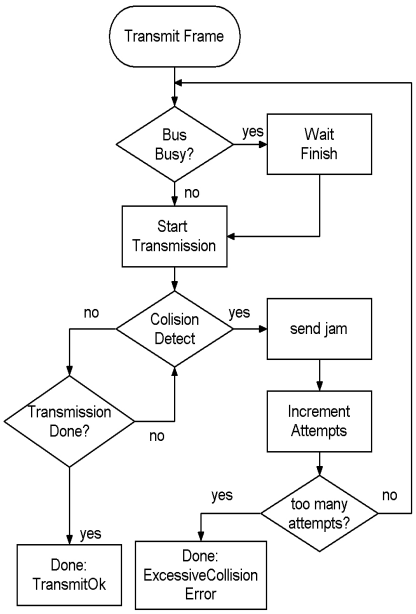

Figure 3.8: CSMA/CD with h-BEB collision resolution algorithm.

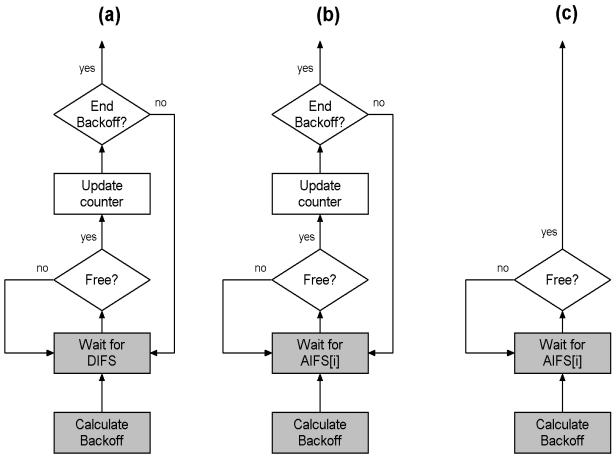

Figure 3.9: Backoff procedures.

mode (Figure 3.9 (b)), the backoff procedure is similar to the DCF one, except that each station has multiple access categories and, for each access category, it has different values for the IFS and the *CW* parameters. Thus, whenever the network is working in the EDCA mode and a collision occurs, all the involved stations will select a locally computed random backoff interval; subsequently, the stations may only retry the transmission whenever the backoff counter for that access category reaches 0. A real-time station implementing the TSm scheme has the same operating behavior as an

EDCA station, except in what concerns the evaluation of the backoff delay and the setting of its IFS values. For the case of the *highest traffic priority* (voice category) transmitted by a TSm station, whenever a frame arrives to the head of the transmission queue, the MAC waits until the medium becomes idle and begins the transmission just after a minimum inter-frame space, *i.e.*, $AIFS[AC\_VO]=DIFS=2\times aSlotTime+aSIFSTime$, with the $CW[AC\_VO]$ parameter set to 0. This behavior guarantees the highest transmitting probability to the TSm stations in a wireless environment shared with multiple unconstrained EDCA stations (*open communication environment*). Any TSm station will always try to transmit its frame in the first empty slot, while all the other EDCA stations will wait during a time interval evaluated by the backoff function.

3.3.3 Properties of the TSm mechanism

The proposed traffic separation mechanism (TSm) is able to prioritize real-time traffic, as long as there is just one active RT station in the same network domain. It can be used as an underlying traffic separation mechanism that enables the provision of real-time communication services in CSMA-based networks. This Traffic Separation mechanism has been applied to Shared Ethernet networks and to WiFi networks (work described in chapters 4 and 6, respectively), where the use of a virtual token passing procedure among independent TSm enabled stations allows the coexistence of multiple RT stations in the same network segment.

It is worth noting that, the target of the proposed Traffic Separation mechanism is to prioritize real-time traffic in heterogeneous CSMA environments, where the set of ST stations is out of the sphere-of-control of the real-time communication architecture. This means that, it is not necessary to control *all* the communication devices in order to support real-time communication. The TSm mechanism enables the prioritization of the real-time traffic, by controlling just the subset of RT stations. This is a relevant enhancement when compared with the traditional approaches to support real-time communication that require the control of *all* the communicating devices.

3.4 Simulation study of the proposed TSm mechanism

In order to validate the proposed TSm mechanism, two simulation models were built for, respectively, IEEE 802.11e and IEEE 802.3 networks, operating in heterogeneous environments with both RT and ST stations. Both simulation models were implemented using the *Network Simulator (NS-2)* tool [92].

The performance measures include: throughput, average packet delay and standard deviation (transfer jitter). The *throughput* is the ratio between the total number of successfully transferred packets and the total number of generated packets for each traffic stream. Therefore, it represents the relative throughput. The *average delay* is the average delay required to transfer a packet, measured from the start of its generation at the application layer to the end of the packet transfer. The standard deviation of the average packet delay is related to a fundamental timing parameter: the message transfer *jitter*; it is given by:

$$\sigma = \sqrt{\frac{1}{N-1} \times \sum_{i=1}^{N} (x_i - \bar{x})^2} \tag{3.2}$$

where N is the total number of simulated packets, x_i is the delay of each transferred packet and $\bar{x}$ is the evaluated average packet delay.

3.4.1 Simulation Setup: IEEE 802.11e

The IEEE 802.11e simulation model considers that n standard IEEE 802.11e stations coexist in the same wireless domain with *one TSm*-enabled station (Figure 3.10). The target of the simulations is to assess the timing behavior of the TSm-enabled station, operating in an *open communication environment*. The obtained results for the IEEE 802.11e stations were validated against previous simulation results presented by Ni, Romdhani and Turletti [93].

Two simulation scenarios are analyzed. For both scenarios, the IEEE 802.11e backoff timing parameters are represented in Table 3.4. In the first scenario (*Scenario 1*), there are 8 mobile stations connected in an *ad hoc* topology: 4 source stations and 4 destinations stations. In the set of source

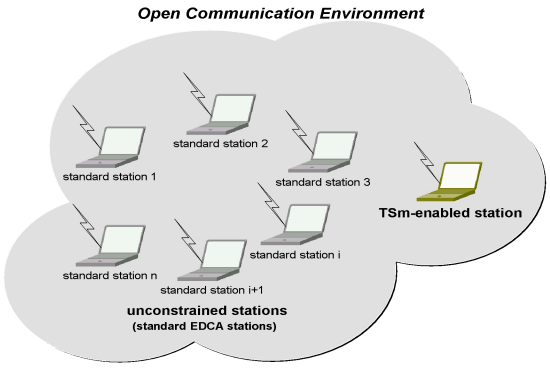

Figure 3.10: Simulation Topology - 802.11e model.

stations, there is 1 real-time station (TSm-enabled) and 3 standard stations ($n = 3$). Each station has a CBR/UDP traffic source with a fixed packet length of 512 bytes operating over IEEE 802.11a PHY mode with a data rate of 36 Mbps. The 3 standard stations equally divide a network load[3] of 70%. The load imposed by TSm-enabled station varies from 1% to 10%, by decreasing the time interval between consecutive packets. Therefore, the total network load varies from 71% to 80%. The objective of this simulation scenario is to analyze the behavior of a TSm-enabled real-time producer station in an environment with an increasing RT traffic load.

Table 3.4: Simulation Scenarios.

Parameters	**RT stations**	**Standard stations**		
	RT traffic	*Voice traffic*	*Video traffic*	*Background*
CW_{min}	0	7	15	31
CW_{max}	0	15	31	1023
AIFSN	2	2	3	7

In the second scenario (*Scenario 2*), the validity of the TSm mechanism is now analyzed in a scenario where *1* TSm-enabled station coexists with a variable number of standard stations (*n=4, 6, 8, 10...20*). All traffics are CBR/UDP sources (the packet sizes shown in Table 3.5 include the whole frame, *i.e.*, data plus header), each station operates at IEEE 802.11a PHY mode and the PHY data rate is set to 36 Mbps. The total network load range

[3] By network load, we mean the aggregate bit rate generated by a set of stations, without considering the MAC and PHY headers. For example, a network load of 10% in a network environment operating at PHY data rate of 36Mbps means that, a set of stations is generating multiple message streams with a total payload of 3600 kbits/s.

varies from 19.17% to 94.37%, by increasing the number of active standard stations. The goal of such simulation scenario is to assess the impact over the RT stations of an increasing traffic load from standard stations. The backoff timing parameters are the same represented in Table 3.4.

Table 3.5: Parameters for each individual station (Scenario 2).

Parameters	**RT stations**	**Standard stations**		
	RT traffic	*Voice traffic*	*Video traffic*	*Background*
Packet Size (bytes)	84	180	1300	1520
Packet Interval (ms)	5	20	16	12.5
Network Load	0.37%	0.2%	1.80%	2.70%

3.4.2 Simulation Results: IEEE 802.11e

The average delay and standard deviation for transferring a packet are represented in Figures 3.11 and 3.12, which show that the real-time traffic transferred by the TSm-enabled station has an average packet delay much smaller than the traffic from standard stations. More importantly, it is clear that, whatever the network load, the average packet delay and the message transfer *jitter* are nearly constant. A significant result is also that the standard deviation of the average delay is almost one order of magnitude smaller than the average delay, which indicates a rather constant value for the average packet delay of real-time traffic. These are important results, as they forecast a predictable communication delay when supporting real-time communications. Additionally, it has been observed that the TSm station did not discard any packet until the simulated network load of 77%.

The average delay for transferring a packet (Scenario 2) is represented in Figure 3.13. The results are intended to compare the impact of external stations (EDCA stations ranging from 4 to 20) upon the average delay for transferring a packet by a TSm-enabled station. The results show in both scenarios that the RT traffic has an average packet delay smaller than other traffics, whatever the number of standard EDCA stations.

The average delay for the RT traffic is no longer constant, when the number of ST stations increases from 4 up to 20. This is due to the increasing number of collisions that occur until the RT messages are able to be transferred, forcing the collision resolution in favor of the RT station.

The average throughput for each type of traffic is plotted in Figures 3.15 and 3.16. It can be seem that throughput is very similar in both

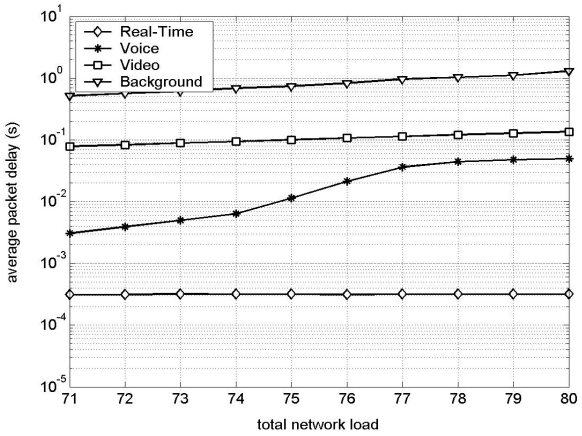

Figure 3.11: Average Delay Scenario 1.

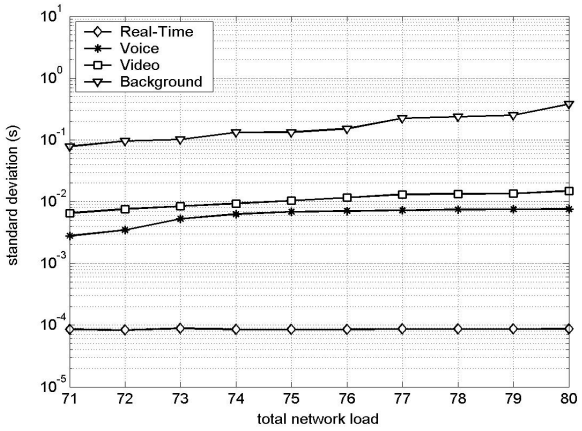

Figure 3.12: Standard Deviation Scenario 1.

voice and video traffics, as the line are almost superposed. However, it is unquestionable the improvement carried on by the TSm mechanism, as for the RT traffic the throughput is nearly constant and equal to 1 (the worst case achieved for the relative throughput value were 0.9963 in the Scenario 1 and 0.9985 in the Scenario 2). This is a remarkable result, as it indicates that even in heavily loaded network scenarios, forcing the collision resolution in favor of the RT station is an adequate methodology to support RT communication in shared broadcast environments.

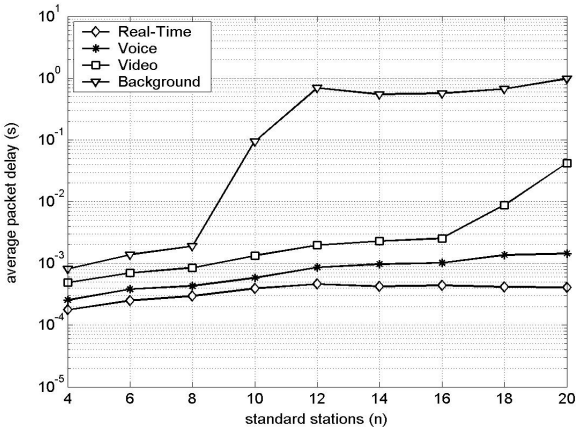

Figure 3.13: Average Delay Scenario 2.

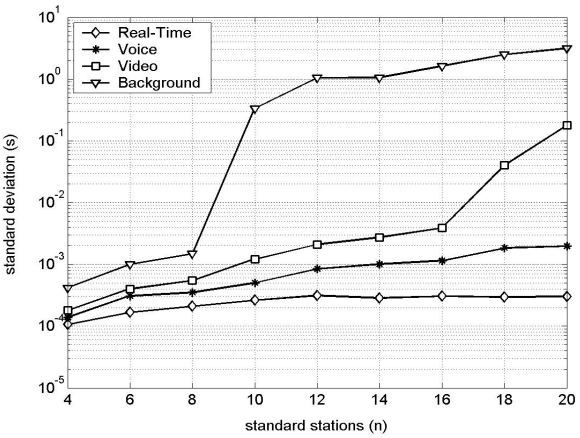

Figure 3.14: Standard Deviation Scenario 2.

3.4.3 Simulation Setup: IEEE 802.3

The IEEE 802.3 (shared Ethernet) simulation model considers a bus topology, where multiple stations are interconnected with a *special station* (Figure 3.17), implementing either the h-BEB (TSm approach) or the BEB algorithms (traditional Ethernet mode). The results were validated against previous simulation results presented by Christensen [94]. The simulation model considers a 10 Mbps Ethernet network, where each station has a Poisson traffic source with a fixed packet length of 250 bytes. The total network load ranges from 40% to 110%. For each simulated load value, 75×10^4 packets are successfully transmitted.

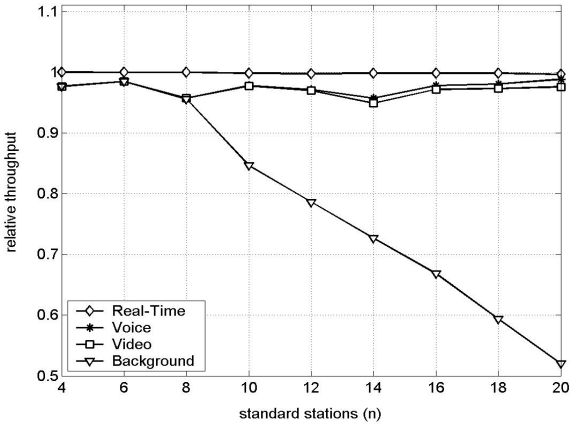

Figure 3.15: Throughput - Scenario 1.

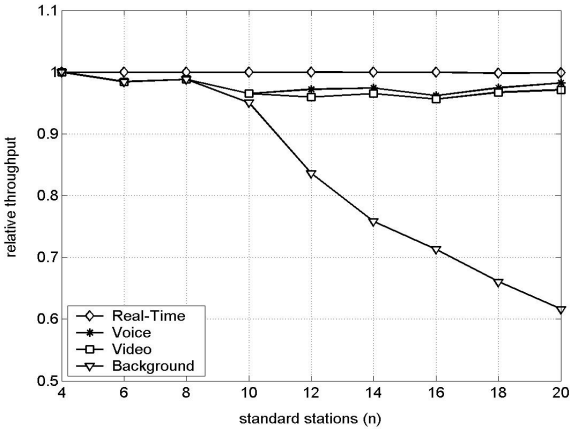

Figure 3.16: Throughput - Scenario 2.

The goal of the presented simulations is to assess the timing behavior of the TSm-enabled station when compared to the standard stations. Two simulation cases are analyzed: the *small population* case that considers 5 interconnected stations imposing a network load ranging from 40% to 110%, where the network load is equally shared by all the stations. In this set of stations, there is one TSm-enabled station (RT station) and 4 IEEE 802.3 standard stations ($n = 4$) and; the *large population* case that extends the small population case to 65 interconnected stations ($n = 64$).

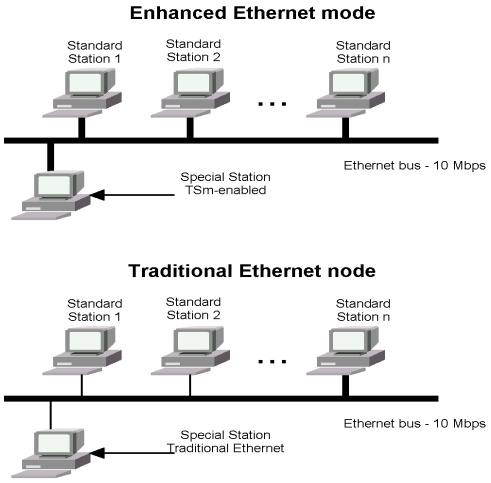

Figure 3.17: IEEE 802.3 simulation scenarios.

3.4.4 Simulation Results: IEEE 802.3

The average throughput for each Ethernet mode is plotted in Figure 3.18 and compared with the theoretical maximum achievable throughput (*i.e.*, without packet collisions or packets being discarded). The average throughput achievable in shared Ethernet networks decreases in the 65% to 110% load region. The difference between the maximum achievable throughput and the obtained throughput (both for the enhanced and the traditional Ethernet modes) is directly related to the lost packets. Similar results have already been shown in previous simulations made by other authors [94, 95]. Moreover, it can be observed that for the small population case the h-BEB station never discarded any packet. This is an expected and important result for a station implementing the TSm approach.

It can be also seen in Figure 3.18 that, when using the TSm-enabled station (Enhanced Ethernet mode) in a small population Ethernet segment, there is a slight increase in the network utilization. This occurs, since the TSm-enabled station is able to transfer almost immediately its own packets, as it is contending to access the communication medium with just 4 other stations.

The average delay for transferring a packet in a small population Ethernet segment is represented in Figure 3.19 for the case of the special station. This Figure shows that for a network load above 50%, the *TSm-enabled station* has an average packet delay much smaller than the *standard station*. For instance, for an 80% network load, the TSm-enabled station takes in

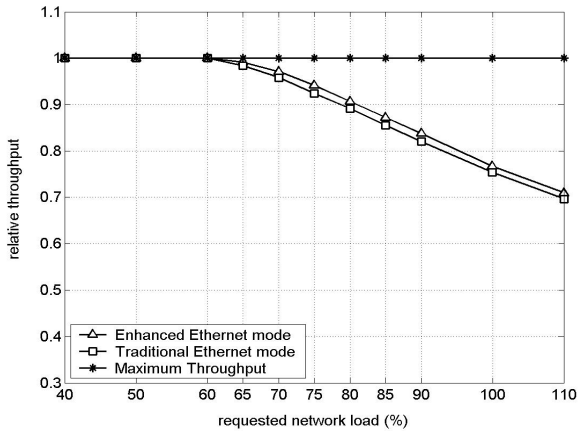

Figure 3.18: Throughput - Small population.

average $0.547ms$ to transfer a packet, while the standard Ethernet station takes in average $56.484ms$. More importantly, it is clear that, whatever the network load, the average packet delay is nearly constant for the TSm-enabled station. This is a very important result, as it forecasts a predictable communication delay when supporting real-time communication.

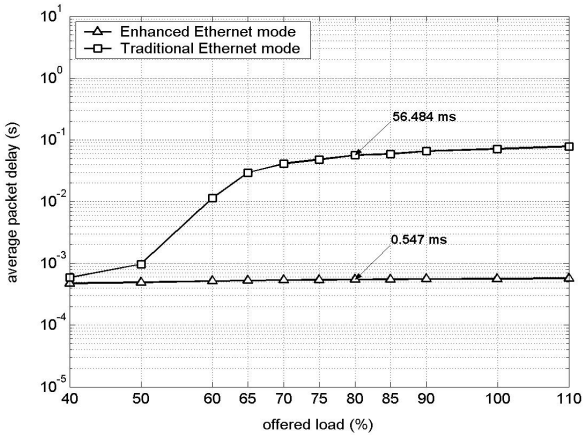

Figure 3.19: Average Delay - Small population.

Finally, Figure 3.20 compares the standard deviation of the average packet delay, which is directly related to the message transfer *jitter*. From this Figure, it becomes clear the difference between the "TSm approach" and the "traditional" Ethernet mode scenarios. For the case of the *special station* implementing the TSm approach, the message transfer *jitter* is nearly constant, whatever the simulated network load. For all standard Ethernet

stations implementing the BEB algorithm, the message transfer *jitter* is of the same order as the average packet delay, which is clearly not adequate for the support of real-time communications.

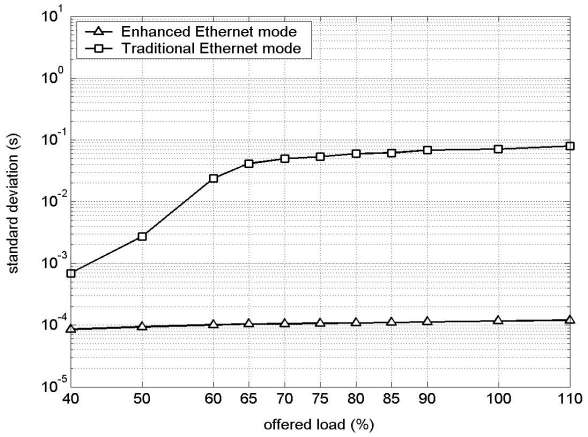

Figure 3.20: Standard deviation - Small population.

Figure 3.21 illustrates the average throughput achievable in shared networks for both the TSm approach (Enhanced Ethernet mode) and the traditional Ethernet mode in the large population case. It can be seem that the throughput is very similar in both scenarios, as the lines are almost superposed. As in the previous case, it has been observed that the TSm-enabled station never discards any packet, whatever the simulated network load.

The average throughput can also be compared with the theoretical maximum achievable throughput, which clearly shows that for a network load above 65%, there is a high rate of packets loss in the network (the difference between the maximum achievable and the obtained throughput are directly related to the lost packets).

The average delay for transferring a packet in a large population Ethernet segment is represented in Figures 3.22 for the case of the special station. As for the small population case, the represented results compare the average delay for transferring a packet in the "TSm approach" *vs.* the average delay for transferring a packet in the "traditional mode". The results illustrated in this Figure shows that the *TSm-enabled station* has also a very small average packet delay. For example, for an 85% network load, the TSm-enabled station takes in average $0.536ms$ to transfer a packet, while a standard station takes in average $800.13ms$.

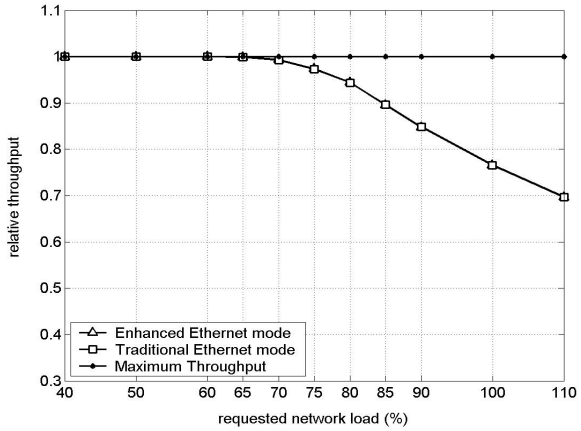

Figure 3.21: Throughput - Large population.

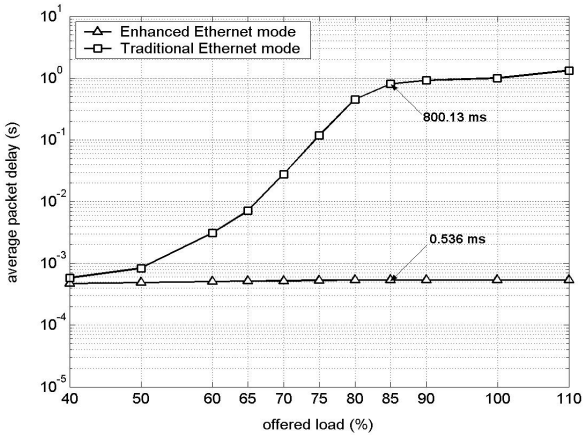

Figure 3.22: Average delay - Large population.

Figure 3.23 compares the standard deviation of the average packet delay. From this Figure, it becomes clear the difference between the "TSm approach" and the "traditional" Ethernet mode scenarios. For the case of the *special station* implementing the TSm approach, the message transfer *jitter* is also nearly constant for the large population case, whatever the simulated network load.

Both results for the small and the large population scenarios clearly show that, whatever the network load or the network population, the average packet delay is nearly constant for the TSm-enabled station in the enhanced network case scenario. These are very important results, as they forecast predictable communication delays for the TSm-enabled station in

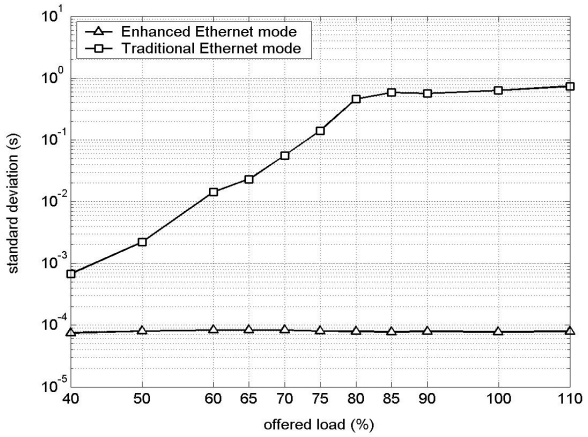

Figure 3.23: Standard deviation - Large population.

very different load and population scenarios. It also demonstrates the effectiveness of forcing the collision resolution in favor of the RT station in communication environments shared with timing unconstrained stations.

3.5 Analytical Study of the IEEE 802.3 vs. TSm

Multiple mathematical models have been developed to represent the behavior of standard Ethernet networks; however, due to the inherent complexity of the mathematical models, several abstractions are usually made. Boggs *et al.* [95] present an interesting survey of performance studies addressing Ethernet standard networks. Other relevant works can be also referred [57, 96, 97, 98]. In this section we present some results for the performance analysis of the h-BEB collision resolution algorithm, and compare them with results obtained from the traditional BEB algorithm.

One of the first Ethernet performance analysis was presented in [99], where the authors draw up a set of equations to perform the exact analysis in heavily loaded Ethernet networks. In that analysis, a constant retransmission probability on each slot has been assumed, and the successful retransmission probability (on the next slot) has been considered to be equal to a constant: p. Therefore, for the case of K active hosts (hosts with packets ready to be transmitted), the probability that only one host will transmit in the beginning of a slot (thus avoiding a collision) is:

$$A = K \times p \times (1-p)^{K-1} \tag{3.3}$$

This probability A is maximized when p=$1/K$. (equal probability of successful retransmission). Such assumption is an interesting approximation for the real backoff function, as it has been shown in multiple simulation studies (*e.g.* [100, 101]). Thus,

$$A = (1 - \frac{1}{K})^{K-1} \tag{3.4}$$

The probability that a host will wait during just 1 slot is $A(1-A)$, while the probability that the contention interval will be exactly n slots is:

$$P(n, K) = A \times (1-A)^{n-1} \qquad n >= 1 \tag{3.5}$$

where n is the collision round number.

Equation (3.5) indicates the probability that the contention interval will be exactly n slots, when all the stations implement the BEB collision resolution algorithm. Furthermore, it is considered saturation conditions, *i.e.*, all hosts have packets ready to be transmitted. Therefore, the probability that one station (*special station*) will win the collision resolution in n collision rounds is:

$$P(n, K) = \frac{A \times (1-A)^{n-1}}{K} \qquad n >= 1 \tag{3.6}$$

where K is the total number of stations in the network segment (one *special station* plus $K-1$ standard stations), as all the stations have equal probability to access the communication medium.

The estimated number of stations trying to transmit is truncated to 1023. Truncating imposes an upper bound to the time interval (backoff delay) that any station must wait before trying to transmit again. Therefore, it results on an upper bound of 1024 potential slots for transmission. Such upper bound imposes a maximum number of 1024 stations that can be supported by a half-duplex Ethernet system [101].

The average number of contention slots is given by [99]:

$$Z = \sum_{n=0}^{\infty} n \times A \times (1-A)^n = \frac{1-A}{A} \tag{3.7}$$

One of the assumptions of the performance analysis carried out by Metcalfe and Boggs [99] is that each station transmits with an equal probability $p = 1/K$. Obviously, this assumption is not suitable for the analysis of the h-BEB algorithm, as in the h-BEB case one of the stations (the TSm-enabled station) transmits at an higher probability. Therefore, a new and adequate set of equations must be devised to perform the probabilistic analysis of the h-BEB collision resolution algorithm.

Theorem: Considering the case of K stations with packets ready to be transmitted (saturation conditions) connected to a shared Ethernet network, where one station referred as *special station* implements the h-BEB algorithm and all other stations implement the BEB algorithm. Following an initial collision, the probability of *special station* winning the collision resolution is given by:

$$P(n,N) = \sum_{j=0}^{N} (-1)^j \left(\begin{array}{c} N \\ j \end{array} \right) \times 2^{-jn} \tag{3.8}$$

where the binomial coefficients are given by:

$$\left(\begin{array}{c} N \\ j \end{array} \right) = \frac{N!}{j!(N-j)!} \tag{3.9}$$

Proof. Considering that N is the number of BEB stations in the network, $K = N + 1$ and n is the number of collision rounds. Firstly, consider the case of $K = 2$, *station A* and *station B* implementing, respectively, the h-BEB and the BEB algorithms. Following an initial collision and considering that both stations have messages ready to be transmitted, the probability of *station A* winning the collision resolution is computed as follows: in the first collision ($n = 1$) the probability of *station A* to transmit is 1/2, as *station A* will always try to transmit in slot 0, while *station B* will backoff during 0 or 1 slot times (according to the BEB algorithm). In the case of a second collision ($n = 2$), *station B* will wait 0, 1, 2 or 3 slot times, while *station A* will try to transmit in slot 0 again, but now with a transmission probability of 3/4 (the probability of a new collision is just 1/4).

Therefore, the probability that *station A* wins the collision resolution is:

$$P(n,1) = \frac{2^n - 1}{2^n} = 1 - 1 \times 2^{-n} \tag{3.10}$$

Lemma: Considering the particular case of 2 stations with packets ready to be transmitted (saturation conditions) connected to a shared Ethernet network, where one station referred as *special station* implements the h-BEB algorithm and all other stations implement the BEB algorithm. Following an initial collision, the probability of *special station* winning the collision resolution is given by $P(n,1) = 1 - 2^{-n}$, $\forall n$.

Proof. It is convenient to rewrite Equation (3.10) as a recurrence relation.

$$\begin{cases} P(0) = 0 \\ P(n) = P(n-1) + \frac{1}{2^n} \end{cases} \tag{3.11}$$

Now, consider that $P(n) = 1 - 2^{-n}$ is true until $n-1$, *i.e.*, $P(n-1) = 1 - 2^{-(n-1)}$. We thus have:

$$\begin{aligned} P(n) &= P(n-1) + \tfrac{1}{2^n} \\ P(n) &= 1 - 2^{-n+1} + \tfrac{1}{2^n} \\ P(n) &= 1 - \tfrac{2}{2^n} + \tfrac{1}{2^n} \\ P(n) &= 1 - 2^{-n} \end{aligned} \tag{3.12}$$

□

Consider now the case of $K = 3$ stations connected to the shared Ethernet network: *station A* with h-BEB algorithm and *station B* and *station C* implementing the BEB algorithm. The probability of *station A* winning the collision resolution is computed as follows: in the first collision (n =1) *station A* tries to transmit in slot 0, while *station B* and *station C* will backoff during 0 or 1 slot times (according to the BEB algorithm). Thus, the possible backoff numbers for the three stations are:

$$\begin{array}{ll} 0,0,0 & 0,0,1 \\ 0,1,0 & \underline{0,1,1} \end{array}$$

and the probability of *station A* winning is 1/4. In the case of a second collision ($n = 2$), *station B* and *station C* will wait 0, 1, 2 or 3 slot times, while *station A* will try to transmit in slot 0 again. Thus, the possible backoff numbers are:

$$\begin{array}{cccc} 0,0,0 & 0,1,0 & 0,2,0 & 0,3,0 \\ 0,0,1 & \underline{0,1,1} & \underline{0,2,1} & \underline{0,3,1} \\ 0,0,2 & \underline{0,1,2} & \underline{0,2,2} & \underline{0,3,2} \\ 0,0,3 & \underline{0,1,3} & \underline{0,2,3} & \underline{0,3,3} \end{array}$$

and the probability of *station A* winning is 9/16. In the case of a third collision ($n = 3$), *station B* and *station C* will wait 0, 1, 2, 3, 4, 5, 6 or 7 slot times, while *station A* will try to transmit in slot 0 again. Thus, the probability of station A winning is now 49/64. Therefore[4], the probability that station A wins the collision resolution is:

$$P(n,2) = \frac{2^{2n} - 2^{n+1} + 1}{2^{2n}} = 1 - 2 \times 2^{-n} + 1 \times 2^{-2n} \tag{3.13}$$

Generally, the probability of the h-BEB station winning the collision resolution in n collision rounds is:

$$P(n,3) = \frac{2^{3n} - \{2^{2n} + 2^n \times (2^n - 1) + (2^n - 1) \times (2^n - 1)\}}{2^{3n}} \tag{3.14}$$

$$P(n,3) = 1 - 3 \times 2^{-n} + 3 \times 2^{-2n} - 1 \times 2^{-3n} \tag{3.15}$$

Applying the same reasoning to the 4 and 5 BEB stations scenarios, the probability of the h-BEB station to win the collision resolution in n rounds would be:

$$P(n,4) = 1 - 4 \times 2^{-n} + 6 \times 2^{-2n} - 4 \times 2^{-3n} + 1 \times 2^{-4n} \tag{3.16}$$

$$P(n,5) = 1 - 5 \times 2^{-n} + 10 \times 2^{-2n} - 10 \times 2^{-3n} + 5 \times 2^{-4n} - 1 \times 2^{-5n} \tag{3.17}$$

Observing equations (3.10), (3.13), (3.15), (3.16) and (3.17), it is clear that the equation coefficients of $P(n,N)$ make up a Pascal's triangle, as follows:

[4] For the sake of simplicity, it is not illustrated the next possible set of backoff values. For further detail consult [17].

$$\begin{array}{l} 1\ 1 \\ 1\ 2\ 1 \\ 1\ 3\ 3\ 1 \\ 1\ 4\ 6\ 4\ 1 \\ 1\ 5\ 10\ 10\ 5\ 1 \end{array} \tag{3.18}$$

Therefore, the general expression for $P(n, N)$ is given by Equation (3.8):

□

3.5.1 Analytical Results: IEEE 802.3 vs. TSm

A comparative analysis of the traditional CSMA/CD protocol *vs.* the proposed traffic separation mechanism have been performed, considering a shared Ethernet environment operating at 10 Mbps. Two cases are analyzed: the *small population* case that considers the case of 4 standard Ethernet stations interconnected (N=*4*) with a *RT station* implementing either the h-BEB (*enhanced Ethernet mode*) or the BEB (*traditional Ethernet mode*) collision resolution algorithms and the *large population* case that extends the small population case to 65 interconnected stations (N=*64*).

In order to have comparable results, the performed analysis compares the results obtained from Equation (3.6) (according to the Metcalfe and Boggs analysis) with the results obtained from Equation (3.8).

Two sets of results are analyzed: In the first, it is represented the probability of transmission for the *special station* after n collision resolution rounds, for both the traditional and the enhanced Ethernet modes. Figures 3.24 and 3.25 illustrate the results of such analysis. It becomes clear that, the *special station* has a much higher transmission probability in the enhanced Ethernet mode than in the traditional mode (as it was expected).

In the second set of results, it is represented the *network accessibility*, that is, the probability that the contention interval will be exactly n slots. In this case, results from Equation (3.5) are directly compared with results obtained from Equation (3.8), as while in the traditional Ethernet mode any station can access the communication medium, in the enhanced Ethernet mode, the h-BEB station will always win the contention. Therefore, in the enhanced Ethernet mode, the probability that the h-BEB station will be able to access the communication medium after n collision resolution rounds is equal to the probability that the contention interval will be exactly

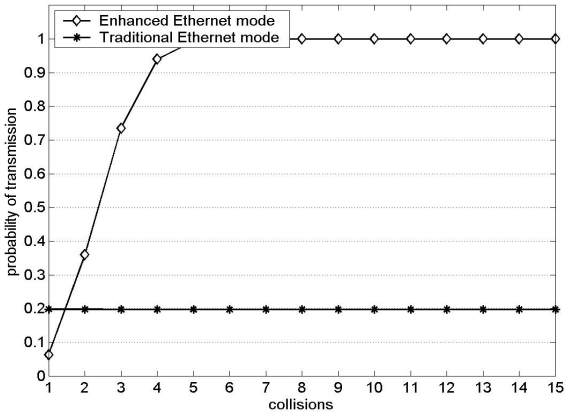

Figure 3.24: Transmission probability for the special station (small population).

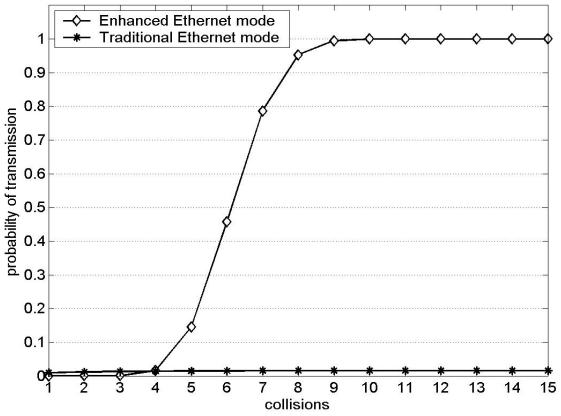

Figure 3.25: Transmission probability for the special station (large population).

n slots. Figures 3.26 and 3.27 illustrate the *network accessibility* for both the traditional and the enhanced Ethernet modes. From these results, it becomes clear that in the enhanced Ethernet mode, the *network accessibility* is smaller than in the traditional mode, for the initial collision resolution rounds. These are expected results, as in the enhanced mode, the special station does not allow any other station to transmit, while it has not succeed to transfer its packets. Therefore, the contention period will be longer than in the traditional mode, whenever the *special station* has packets to be transferred.

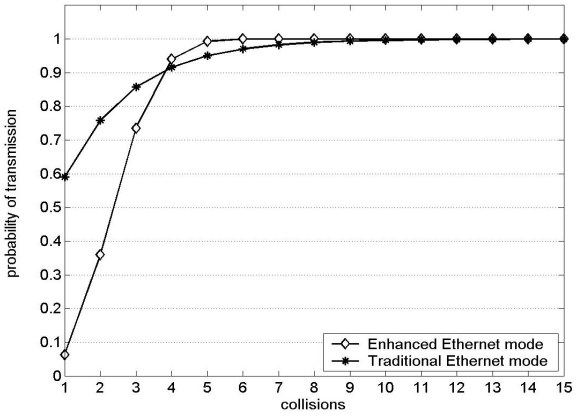

Figure 3.26: Network accessibility (small population).

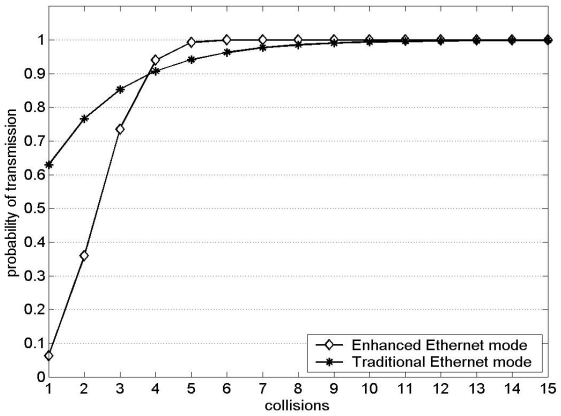

Figure 3.27: Network accessibility (large population).

3.6 Summary

In this chapter we describe a new Traffic Separation mechanism (TSm) that allows the coexistence of CSMA standard stations with enhanced (real-time) stations in the same network domain. The TSm mechanism is able to force the collision resolution in favor of the real-time station in CSMA networks, when the communication medium is shared with timing unconstrained traffic. A simulation analysis of the TSm mechanism has been done. The performance measures included: relative throughput, average packet delay and standard deviation of the average packet delay.

The obtained results for IEEE 802.11 wireless networks demonstrate that

the proposed underlying TSm mechanism guarantees the highest transmitting probability for the TSm station in a wireless environment with multiple EDCA standard stations. More importantly, it is clear that, when varying the real-time traffic network load, both the average packet delay and related standard deviation for the real-time traffic are nearly constant. Furthermore, the results obtained in both analyzed scenarios show that the TSm mechanism has a good performance in an *open communication environment*, where real-time TSm-enabled stations coexist with multiple IEEE 802.11e standard stations generating timing unconstrained traffic.

The obtained results for IEEE 802.3 wired networks also demonstrate that the TSm mechanism guarantees an average access delay significantly smaller for the TSm-enabled station, when compared with the access delay for the standard Ethernet stations, in small and large population scenarios. For instance, for an 80% network load and a small population scenario, the TSm-enabled station takes in average $0.547ms$ to transfer a packet, while the standard Ethernet station takes in average $56.484ms$.

Additionally, a probabilistic analysis was also performed for IEEE 802.3 wired networks in a heavily loaded network scenario. The analysis results show that an h-BEB station has a significantly higher probability to send a message up to the i^{th} collision round than any BEB station.

The obtained results (both by simulation and by probabilistic analysis) are very important, as they forecast a predictable communication delay when supporting real-time traffic in communication environments shared with timing unconstrained traffic. We are convinced that the proposed traffic separation mechanism will enable the provision of real-time communication services in CSMA-based networks. Specially when there are communicating devices out of the sphere-of-control of the RT-communication architecture, as it is always the case in wireless environments. Therefore, it enables the setup of a simple "forcing collision resolution" scheme in favor of RT stations, able to be implemented using COTS hardware (compatibility level 3).

Chapter 4

The VTPE-hBEB architecture

The previous chapter presented a traffic separation mechanism (TSm) that enables the provision of real-time communication services in CSMA-based networks. This chapter presents and evaluates a shared Ethernet deterministic architecture able to interconnect multiple sensors, controllers and actuators at the field level of an Ethernet-based industrial communication architecture. The proposed mechanism, referred as VTPE-hBEB, is based on a virtual token that is circulated among enhanced stations, complemented by the TSm traffic separation mechanism. One of the main advantages of the proposed solution is that, it allows the coexistence of multiple enhanced (real-time) devices with Ethernet standard (non real-time) devices in the same network domain, without controlling the traffic behavior of the latter. That is, it allows the implementation of real-time communication approach of compatibility level 3. This chapter is largely drawn from the following published work: "Real-Time Communication in Unconstrained Shared Ethernet Networks: The Virtual Token-Passing Approach" (Carreiro et. al. [18]).

4.1 Introduction

The Computer Integrated Manufacturing (CIM) is a widely used reference model for the industrial automation hierarchy. It is typically composed of up to five levels. The lowest level of the CIM model is usually referred as the *field* level. Since the early 1990s, multiple fieldbus network technologies have been proposed to interconnect sensors and actuators to controllers at the

field level. For a long a time, a common characteristic at this level was the need for transferring just small sized packets at low data rates. However, such a common characteristic is no longer applicable. Since the last 5-8 years, it is quite common the requirement to support the transmission of video or voice data, thus reaching the former LAN domain [2].

Within this context, Ethernet has emerged as a *de facto* communication standard for all levels of the automation systems. This is a consequence of the widespread availability of COTS (hardware/software) communication components and, the promising integration possibility into the company LANs at higher levels of the automation hierarchy. Although Ethernet on industrial environments remain disputed by many companies, the term *Industrial Ethernet* is commonly used as a reference to all those solutions that use the Ethernet protocol in an industrial environment.

An usual classification of the automation communication requirements is based on the delivery time, which may be divided into three classes [2, 6]:

- Firstly, a low-speed class for human control with delivery times around $100ms$. This timing requirement is typical for the case of humans involved in the system observation (10 pictures per second can already be seen as a low-quality movie), for engineering and for process monitoring.

- Secondly, a process control class, with delivery time requirement below $10ms$. These are the requirements for most tooling machine control systems like PLCs or PC-based control.

- Thirdly, the most demanding class is related to motion control applications, where the requirements for delivery time are below $1ms$.

When Ethernet networks started to be used to support automation systems, its main drawback was the CSMA/CD medium access protocol. Such medium access protocol was not able to ensure the real-time delivery of time-critical data. Consequently, several commercial companies have developed extensions to the legacy shared Ethernet standards that fulfilled the real-time Ethernet specifications. For example: *Profinet*, *EtherNet/IP*, *EtherCAT*, *Ethernet Powerlink* and *Modbus*. Most part of these extensions recommend the use of Ethernet Switching hubs that micro-segment the network, preventing the occurrence of collisions between frames.

Profinet is the Ethernet-based automation standard maintained by PROFIBUS International and more than 50 companies (including Siemens). In 2003 was ratified as the International Standard IEC 61158 and IEC 61784. According to Feld [102], Profinet version $v2$ provides a reaction time in the range of 5-10 ms. In the most recent *Profinet* versions ($V2$ and $V3$), a middleware-scheduling layer provides an adequate priority scheme to the real time data. Class 3 communication is reached in version $V3$ with a TDMA-based scheduling based on a communication ASIC (Application Specific Integrated Circuit), where a time slot is exclusively reserved for real-time communication within the communication cycle [103].

EtherNet/IP is the industrial communication standard originally defined by Rockwell. It is supported by the Open DeviceNet Vendor Association (ODVA) and by ControlNet International. It makes use of an open application layer protocol, which is based on Control Information Protocol (CIP) that is used in both *DeviceNet* and *ControlNet*. This topology implement a common set of services at the network level, where all the devices organize their data into a common object model. The CIP family of protocols contains a fairly large collection of commonly defined objects [104]. In the most recent Ethernet/IP specification, real-time messages are processed at the highest priority by the switches. Such Ethernet/IP specifications satisfy class 2 applications.

EtherCAT is an open communication technology, for which there is an on-going effort for an IEC standardization. It sets a new standard for real-time performance using twisted pair cables or fiber optic cables, and it supports line, tree or star topologies. With *EtherCAT*, the data exchange is fully based on a pure hardware machine, over a logical ring structure, where a master clock determines the propagation delay. External synchronization is based on the IEEE 1588 standard [105]. *EtherCAT* has different addressing options for different types of communication. It is optimized for each particular requirements [106]. Basically, it employs a master/slave medium access, where the master node sends a frame to the slaves, which extract and insert data into these frames. Class 3 communication is also reached using EtherCAT solutions.

Ethernet Powerlink protocol is based on the standard IEEE 802.3 layers. Deterministic time is achieved applying a cyclic timing schedule to all the connected nodes. The schedule is divided in isochronous and asynchronous phase. Time-critical data is transferred during the isochronous phase. The asynchronous phase reserves bandwidth for non time-critical

data. The node management grants the access to the physical medium via the exchange of an explicit message (token), thereby preventing collisions. The Ethernet Powerlink Standardization Group (EPSG) recommends the use of repeater hubs instead of switching hubs within the real-time domains. Class 3 communication is easily reached by Powerlink.

Modbus protocol, developed by Modicon in 1979, is based on master-slave/client-server communication between devices. It is a protocol positioned at level 7 of the OSI model. It defines a simple protocol data unit (PDU), that is independent of the underlying communication layers. The Modbus messaging communication uses four type of messages: a *Modbus Request* is the message sent on the network by the client to initiate a transaction; a *Modbus Indication* is the request message received on the server side; a *Modbus Response* is the Response message sent by the Server; a *Modbus Confirmation* is the response message received on the client side. Modbus just fulfills the class 1 requirements.

As referred before, most industrial Ethernet solutions are implemented using Ethernet switching hubs, that uses full-duplex capabilities of Ethernet. However, one of the main disadvantages of switched Ethernet is that the switch output buffers can be easily exhausted if bursts of messages are sent to the same output port. This situation leads to the loss of messages and it can occurs more often than desired. For example, in distributed control systems, the producer/consumer model is typically used. According to this model, one producer of a given datum (*e.g.* a sensor reading) send it to its several consumers. This model is efficiently supported in Ethernet by means of special class of addresses, the multicast addresses. Each network interface card can define a local table with the multicast addresses that it should receive. However, as the switch has no knowledge of such local tables, it treats all the multicast traffic as broadcast traffic, *i.e.*, messages with multicast destination addresses are transmitted to all ports.

Another relevant drawback of switch-based solutions is related to the topology. The simplest topology is the star topology, because it only has one switch. However, the star topology requires a maximum cabling, which is barely adequate for automation systems. For this reason a line topology is commonly used (*e.g.* Profinet), where there must be one switch per device [39]. Therefore, the number of switches between data producers and consumers is typically much larger in an automation environment than in an office environment. Therefore, a special attention has to be taken to the delay introduced by the cascaded switches [107].

The Ethernet Powerlink (EPL) [30] is a relevant exception in the Industrial Ethernet market. Despite a recent version (version 2.0) that also allows operation over Switched Ethernet networks, the Ethernet Powerlink Standardization Group (EPSG) recommends the use of 100BaseTX/FX Ethernet repeater hubs instead of switching hubs within the real-time domains. The intention is to minimize path delays and frame jitters. According to Sauter [108], Ethernet Powerlink solutions were the first available Industrial Ethernet on the market that fulfilled the class 3 requirements.

Traditionally, the RT communication behavior in shared Ethernet environments has been guaranteed through the tight control of every communicating device [5]. The coexistence of RT controlled stations with timing unconstrained stations has been made possible by constraining the traffic behavior of the latter. That is one of the main drawbacks of state-of-the-art real-time networks when dealing with next generation communication environments[1].

The main target of this chapter is to propose a shared Ethernet deterministic architecture with a medium access control scheme similar to Ethernet Powerlink, where the explicit token passing is replaced by a virtual token approach. The main advantage of the proposed approach is that contrarily to Ethernet Powerlink, it allows the coexistence of Ethernet standard (non real-time) devices with multiple enhanced (real-time) devices in the same network segment. This means that, instead of controlling *all* the traffic generated by *all* the stations, the proposed mechanism will control only the traffic generated by the stations supporting RT-traffic, prioritizing such traffic over the unconstrained multipurpose traffic.

This chapter is organized as follows: Section 4.2 shows how VTPE-hBEB mechanism works, highlighting how the enhanced stations are able to transfer real-time traffic within unconstrained traffic environments. Afterwards, a timing analysis of the VTPE-hBEB mechanism is presented in section 4.3, showing that the achieved timings are adequate for the target applications. In section 4.4, the VTPE-hBEB implementation is briefly described. It is worth mentioning that, the VTPE-hBEB proposal has been developed in cooperation with Institute of Electronics and Telematics Engineering of Aveiro (IEETA) at the University of Aveiro and, all hardware implementations presented in this chapter were exclusively built by the IEETA group members. Afterwards, in section 4.5, the main results from the performance

[1] This issue was extensively discussed in chapters 1 and 2.

analysis done using the VTPE-hBEB test-bed are presented. Finally, some conclusions are given.

4.2 The VTPE-hBEB architecture

The main target of the VTPE-hBEB architecture is to enable the coexistence of unconstrained default (non real-time) Ethernet stations with enhanced (real-time) stations in the same real-time communication infrastructure, imposing higher priority to the real-time traffic. The proposed VTPE-hBEB architecture is based on the control of the medium access right, by means of a Virtual Token Passing (VTP) procedure, complemented by the TSm mechanism (described in chapter 3). The TSm mechanism guarantees that, whenever a real-time station is contending for the medium access, it will win the contention prior to any default station.

The TSm mechanism consists in the use of backoff interval equal to zero for RT stations[2]. This behavior guarantees the highest transmitting probability to the VTPE-hBEB station, as it will always try to transmit its frame in the first empty slot, while all the other stations will wait between 0 and $2^n - 1$ slot times. Nevertheless, whenever two or more VTPE-hBEB stations simultaneously contend for the medium access, they will collide and eventually discard the frame (after the maximum number of retransmission attempts). This behavior is overcome by means of a Virtual Token Passing (VTP) procedure, that serializes the VTPE-hBEB stations.

The VTPE-hBEB architecture considers a process group G with np members. The membership is represented as $L = \{NA_1, NA_2,..., NA_{np}\}$. The notion NA_i denotes the *i-th* station in G and it is also used as station identification (ID) for NA_i itself in this chapter. The VTPE-hBEB architecture circulates a virtual token in L. Differently from the well known token bus protocol, the proposed virtual token is not physical. Specifically, all members of group G maintain an Access Counter (ACo). The generic *i-th* VTPE-hBEB station *captures* the virtual token when the value of its ACo counter equals NA_i.

Whenever, a token holding station has queued messages, it will immediately transfer one RT message. The TSm mechanism guarantees that the VTPE-hBEB station will win the medium access contention. Whenever a

[2] In this chapter RT stations are also referred as VTPE-hBEB stations.

successful frame is transferred, an interrupt occurs simultaneously in all RT stations. This interrupt event is used to increment each of the local *ACo* counters[3]. On the other hand, if the token holding station does not have any message to be transferred, it will allow Ethernet (non real-time) standard stations to contend for the bus access, during a time interval t_{guard} ($t_{guard} = 2 \times InterFrameGap$). If the bus remains idle during t_{guard}, an interrupt will be generated in all the RT stations and all the *ACo* counters will be incremented, which corresponds to an implicit token passing.

Figure 4.1 illustrates the VTPE-hBEB mechanism, that is represented by four procedures: *Initialization*, *Main*, *Transmission* and *Listening*.

```
1  ∀ process NA_i
2  Initialization: ACo ← NA_1; set(np); reset(t_guard); c_reset ← 0;
3  Main: while c_reset ≤ RN do
4            if AC = NA_i then
5               {wait InterFrameGap};
6               NA_i executes Transmission;
7            else
8               {wait InterFrameGap};
9               NA_i executes Listening;
10           endif
11       end while
```

Figure 4.1: VTPE-hBEB mechanism.

1. *Initialization* procedure: the *ACo* counter value is set to 1 in all VTPE-hBEB stations, and the number of VTPE-hBEB stations is preset to *np*. Besides, two other variables (t_{guard}, c_{reset}) are defined: t_{guard} is the timer that manages the guard time needed to detect if the VTPE-hBEB node holding the token do not have anything to transmit. The variable of type integer c_{reset} is a special collision counter used for re-initialization purposes;

2. *Main* procedure: it is executed at the beginning of each time unit[4], where every VTPE-hBEB station firstly verifies if there were *RN* con-

[3] The *ACo* must be increased by "mod" operation, *i.e.* $ACo = (ACo \bmod np) + 1$. Therefore, when it is referred that the *ACo* is increased, it is executed such "mod" operation.

[4] In this work, it is assumed that each time unit corresponds to discrete times, whose length is equal to the *InterFrameGap* (96 bit times).

secutive collisions ($c_{reset} \leq RN$)[5]. Afterwards, depending on both the channel event during the last time unit and the ACo value, each VTPE-hBEB station will take a specific action;

3. *Transmission* procedure: it is only executed by the VTPE-hBEB station holding the token ($ACo = NA_i$) and, it is initiated after the medium being idle during an $InterFrameGap$;

4. *Listening* procedure: it is also only initiated after the medium being idle during an $InterFrameGap$ and, it is executed by all VTPE-hBEB stations that do not have the token and by the VTPE-hBEB station holding the toking that does not have any message to be transferred.

As mentioned before, according to both the channel event during the last time unit and the ACo value, each VTPE-hBEB station will take a specific action. Thus, five channel states are defined (determined at the beginning of each time unit):

1. channel *busy*: One message is being transmitted over the bus channel.

2. $successful\_transmission$ from other stations: The bus channel is idle, and a successful message transmission from other station finished one time unit ago.

3. channel *continuing_idle*: The bus channel is idle, and it was also idle one time unit ago.

4. channel *collision*: Two or more messages were transmitted over the bus channel one time unit ago.

5. $successful$ transmission: The channel is idle, "I am the transmitting station", and I finished the transmission of one message.

[5] In fact, the TSm mechanism (described in chapter 3) solve collisions in a bounded time interval or, it eventually discards the message. The special counter c_{reset} is used to prevent the case when the ACo counters become unsynchronized. The *Initialization* procedure (reset mechanism) that synchronizes all ACo counters is invoked whenever there is no successful frame transmission during t_{col} and, it is not detected a bus idle with duration greater or equal than t_{guard} (when $c_{reset} = RN$). Summing up, if the ACo counters become unsynchronized, there will be RN (maximum number of transmission attempts) consecutive collisions, always in the first empty slot. In such a case, the VTPE-hBEB mechanism must be restarted, executing the *Initialization* procedure.

Firstly, whenever the VTPE-hBEB station *captures* the virtual token ($ACo = NA_i$), it will execute the *Transmission* procedure (Figure 4.2). This procedure works as follows. If the VTPE-hBEB station holding the token have a RT message to transfer, it will immediately start the transmission. If a collision occurs, it will retry the transmission until the maximum defined transmission number of attempts (RN). Whenever, a successful transmission occurs, the VTPE-hBEB station holding the token will increase its ACo value, passing the virtual token to the next station. Conversely, whenever the VTPE-hBEB station holding the token does not have any RT message to transfer, it will executed the *Listening* procedure (line 16, Figure 4.2).

```
1  Transmission:
2  if NA_i has a message to be transmitted then
3      start the transmission; wait for transmission to complete;
4      if successful transmission then
5          Done : TransmitOK; ACo++;
6          reset(t_guard); c_reset ← 0;
7      else
8          c_reset++;
9          if c_reset > RN then
10             Done : ExcessiveCollisionError; go to Initialization;
11         else
12             go to Transmission;
13         endif
14     endif
15 else
16     go to Listening;
17 endif
```

Figure 4.2: Transmission procedure.

As illustrated in Figure 4.1, all VTPE-hBEB stations that do not have the token ($ACo \neq NA_i$) will execute the *Listening* procedure (Figure 4.3) and, depending on the channel state, these VTPE-hBEB stations take a specific action:

1. channel *busy* (lines 5-6): All VTPE-hBEB stations wait for the end of the current transmission;

2. $successful\_transmission$ from other stations (lines 7-9): All VTPE-hBEB stations increment its AC value, reset t_{guard} and update t_{reset};

3. channel *continuing_idle* (lines 10-14): All VTPE-hBEB stations update and verify timer t_{guard}. If $t_{guard} \geq 2 \times InterFrameGap$, all VTPE-hBEB stations increment its ACo value. Besides, t_{guard} and c_{reset} are updated;

4. channel *collision* (lines 15-16): All VTPE-hBEB stations update t_{guard} and increment c_{reset}.

```
1 Listening:
2 At the beginning of the next time unit
3   event ← channel event during the last time unit;
4   switch (event)
5       case busy:
6           wait for transmission to complete;
7       case successful_transmission:
8           Done : TransmitOK; AC++;
9           reset(t_guard); c_reset ← 0;
10      case continuing_idle:
11          start(t_guard);
12          if t_guard ≥ (2 × InterFrameGap) then
13              AC++; reset(t_guard); c_reset ← 0;
14          endif
15      case collision:
16          update(t_guard); c_reset++;
17  end switch
```

Figure 4.3: Listening procedure.

Figure 4.4 illustrates an example of the behavior of the proposed VTPE-hBEB mechanism. Assume that 3 VTPE-hBEB stations are sharing the bus with 1 standard Ethernet station. According to the initialization procedure (line 2, Figure 4.1), the access counter (ACo) value is equal to 1 in all VTPE-hBEB stations after *InterFrameGap* (instant *a1*). Therefore, station NA_1 (token holding station) runs the *Transmission* procedure, while stations NA_2 and NA_3 execute the *Listening* procedure[6]. However, as station NA_1 has no message to be transferred, this station will also execute the *Listening* procedure. This procedure consists in continuously monitoring the bus.

[6] When a signal is being transmitted on the Ethernet channel, that condition is called *carrier*. When an Ethernet station wants to transmit a frame, it waits until the channel goes idle, as indicated by an *absence of carrier*. The frame is immediately transmitted whenever there is no carrier and the period of no carrier has continued for an amount of time that equal or exceeds the IFG [109]. Therefore, the presence/absence of carrier signal can be easily used to synchronize all variables defined in the VTPE-hBEB architecture.

According to the example, the next events are preceded by channel idle periods. Therefore, all VTPE-hBEB stations start the timer t_{guard} (line 11, Figure 4.3). Then, timers t_{guard} will expire in all stations at instant *a2* (after $2 \times InterFrameGap = 19.2\mu sec$) and the virtual token is passed to station NA_2 (lines 12-14, Figure 4.3). As the channel remains idle (NA_2 station has no message to be transmitted), timer t_{guard} will expire again at instant *a3*, passing the virtual token to station NA_3. It is interesting to note that between instants $a1$ and $a3$, any ST station could start a frame transmission.

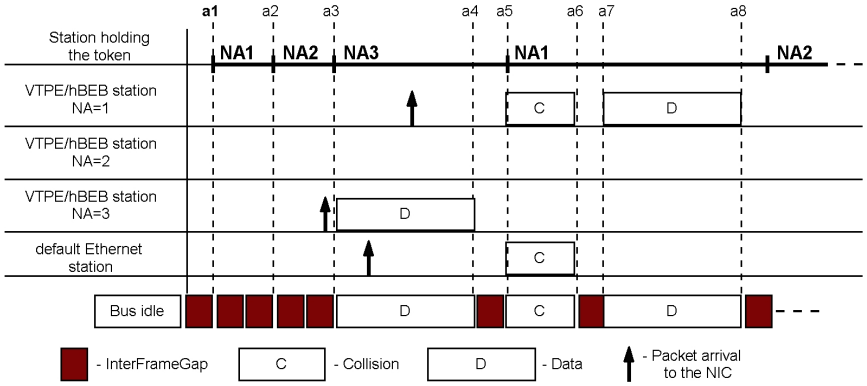

Figure 4.4: Behavior of the VTPE-hBEB.

As the medium remains idle, station NA_3 may start to transfer its own real-time messages. Then, as a successful frame is transferred, an interrupt occurs simultaneously in all RT nodes whenever a successful frame is fully transferred (lines 4-6, Figure 4.2 and lines 7-9, Figure 4.3). Then the virtual token is passed again to station NA_1. Afterwards, considering that between instants *a3* and *a4*, both the ST and RT stations generated one packet, both stations will start the frame transmission at the same time (just after detect the bus idle during an $InterFrameGap$ period). Consequently, a collision will occur and, at instant *a6*, the ST station will select a backoff time according to the BEB algorithm. According to the TSm mechanism, the VTPE-hBEB station will retry the transmission after just the medium being idle during an $InterFrameGap$. This means that NA_1 will start the transfer of its real-time messages at instant *a7*. Afterwards, when the end of transmission is detected (considering that the ST station involved in the collision selected a backoff value greater than 0 and any other ST station has not started its transmission at instant *a7*), the variable ACo is incremented

(lines 4-6, Figure 4.2 and lines 7-9, Figure 4.3) at one InterGrameGap after instant *a8*.

4.3 Timing Analysis

In this section, it is presented the timing analysis of an Ethernet network interconnecting multiple VTPE-hBEB stations with Ethernet standard stations. This analysis clearly illustrates the real-time behavior of the proposed VTPE-hBEB architecture.

Consider a process group G with np VTPE-hBEB stations, $L = \{NA_1, NA_2, ..., NA_{np}\}$, with addresses ranging from 1 to np. Each VTPE-hBEB station accesses the network according to the VTPE-hBEB scheme, *i.e.*, first station 1, then station 2, 3,... until station np, and then again station $1, 2, ...np$. The standard Ethernet stations implement the traditional BEB collision resolution algorithm.

First of all, consider a n-collision scenario. In such a case, the maximum delay to transfer a real-time message, when the VTPE-hBEB station is holding the token, is illustrated in Figure 4.5. According to the VTPE-hBEB scheme, such station transmits its message using the underlying traffic separation mechanism (TSm - chapter 3); that is, it always tries to transmit its message in the first available time slot.

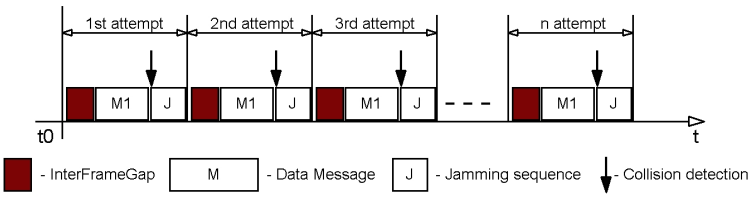

Figure 4.5: A n-collision scenario solved by the TSm mechanism.

Therefore, when a VTPE-hBEB station holding the token has a message ready to be transferred (M), it will wait that the channel becomes idle and, just after a brief Interframe Gap ($InterFrameGap$: 12 byte times) it transmits its message. If a collision occurs during the transfer of the first 64 bytes of message M (M1), a jamming sequence will be broadcasted (J: 4 byte times). Afterwards, the station will wait again during an $InterFrameGap$

and, according to the underlying traffic separation mechanism, it will immediately start to transmit its message. If a second collision occurs[7], a new jamming sequence (J) will be broadcasted and the VTPE-hBEB station will wait again for another $InterFrameGap$, before starting to transmit. The cumulative result from $t0$ up to the beginning of the third attempt is 160 bytes or $128\mu s$ (at a 10 Mbps bit rate). The maximum time (t_{col}) that a VTPE-hBEB station holding the token will wait before starting to transfer a successful message or eventually to discard it is $960\mu s$ (Table 4.1).

Table 4.1: Maximum delay to start transferring a message: VTPE-hBEB collision resolution algorithm.

Retry Number	**Max cumulative delay (SlotTimes)**	**Max delay (μs)**
1	1	64
2	2	128
3	3	192
4	4	256
5	5	320
6	6	384
7	7	448
8	8	512
9	9	576
10	10	640
11	11	704
12	12	768
13	13	832
14	14	896
15	15	960
16	discard frame	

It is clear that the underlying prioritizing algorithm solves collisions in a bounded time, or it eventually discards the message. Therefore, it is of utmost importance to focus on the probability of a message frame being discarded by the VTPE-hBEB algorithm, whenever the number of collision resolution rounds exceeds 15.

Such probability has been analytically evaluated for an highly loaded network scenario, and is equal to 1.22×10^{-4} and 1.95×10^{-3}, respectively for small (5 stations) and large population (65 stations) scenarios [16]. For

[7] A collision only occurs during the transfer of the first 64 bytes of message. If a collision occurs after 64 byte times, then it is considered an error and called a *late collision*. A late collision is a serious error, since it indicates a problem with the network system, and since it causes the frame being transmitted to be discarded [109].

more realistic load scenarios, it has been verified by simulation that a VTPE-hBEB station never discards any packet, whatever the simulated network load (simulation scenario: 75×10^4 h-BEB simulated messages in a 10Mbps network with 64 standard Ethernet stations and one h-BEB station, with an external network load ranging from 40% to 110%) [88]. Such results are consistent with the claim that the VTPE-hBEB mechanism is able to support most part of soft real-time applications, as they confirm a rather small probability of any message being discarded.

Therefore, if it is considered that no message is discarded by the VTPE-hBEB station holding the token, the maximum time that a VTPE-hBEB station holding the token waits to transfer a real-time message is given by:

$$T\_hBEB = t_{col} + InterFrameGap + t_{message} \tag{4.1}$$

where t_{col} is the worst-case delay to start transferring a successful message ($960\mu s$ at 10 Mbps or $96\mu s$ at 100 Mbps) and $t_{message}$ is the time to transfer a message from the VTPE-hBEB station.

On the other hand, when the VTPE-hBEB station holding the token does not have any real-time message ready to be transferred, the standard Ethernet stations in the network segment can try to start transferring their own messages. In such a case, all the VTPE-hBEB stations will wait during a time interval t_{guard}, within which any Ethernet standard station may try to start transferring a message. If the collision resolution round is no longer than t_{col} ($960\mu s$), or if the bus remains idle during a time interval equal to t_{guard}, an interrupt will be generated and all the AC counters will be incremented (*i.e.*, there will be a Virtual Token Passing).

In Figure 4.6 it is exemplified the maximum time interval that a VTPE-hBEB station is allowed to hold the token, even if it does not have any real-time message ready to be transferred.

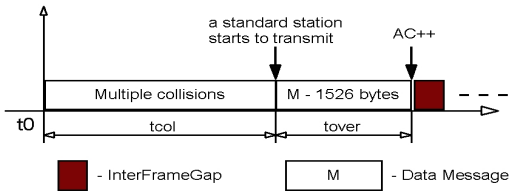

Figure 4.6: Token holding time.

Such a worst-case arises when multiple collisions occur. The time interval t_{col} is long enough to allow a VTPE-hBEB message transfer, as the VTPE-hBEB stations that are not holding the token do not known if the colliding messages are from just Ethernet standard stations, or if there is among the colliding messages, a message from a VTPE-hBEB station that is holding the token. In the latter, the time interval t_{col} guarantees that the VTPE-hBEB station holding the token station will be able to transmit its message and an interrupt will occur when the message transfer is finished. Otherwise, if the collision resolution is not solved during the time interval t_{col}, most probably, the ACo counters become unsynchronized. Therefore, the reset mechanism is invoked (line 2, Figure 4.1) whenever there is no successful transmission during a time interval equal to t_{col} neither the medium becomes idle during a time interval greater or equal than t_{guard}. Then station NA_1 in the logical ring will be able to contend for the medium access. The above described condition occurs when $c_{reset} > RN$ (line 3, Figure 4.1).

The worst-case for the token holding time occurs when the collisions are occurring just among standard Ethernet stations and at instant $(t_{col} - \varepsilon)$, a standard Ethernet station starts to transmit a 1526-byte message (t_{over}), which is the longest message that can be transferred in an Ethernet network.

Therefore, the maximum time that a VTPE-hBEB station may hold the token is given by:

$$T_{TH} = t_{col} + t_{over} \tag{4.2}$$

As the token rotation time is the time interval between two consecutive token visits to a particular station, the worst-case token rotation time, denoted as TRT, is given by:

$$TRT = np \times T_{TH} \tag{4.3}$$

where T_{TH} is as defined in Equation (4.2).

4.4 VTPE-hBEB implementation

The hardware setup specifically built for the assessment of the VTPE-hBEB architecture was implemented at the Institute of Electronics and Telematics Engineering of Aveiro (IEETA), within the context of a collaborative

project. For further details, please refer to [110]. In this section, we briefly describe that hardware setup.

The VTPE-hBEB node implementation can be analyzed in two domains: hardware and software. Considering the former, the current RT node implementation is composed of two similar sub-nodes, each one containing a MCU (MicroController Unit) and an Ethernet controller (among other components). The software running in each sub-node is different; one runs the VTPE-hBEB protocol while the other implements the target application.

The proposed VTPE-hBEB architecture can be implemented using any Ethernet controller that offers *BEB disabling* and *interrupt* support. When this work has been started, only the CS8900A-CQ Ethernet controller [111] was available with BEB disabling support. However, this controller does not support interrupts in 8-bit mode. According to the application note AN181 [112], when working in 8-bit mode, a polling mechanism must be used to access the controller receive event register. Currently, there are several Ethernet controllers able to fullfil these requirements. Some example are the ENC28J60 from Microchip [113], the CP2200-GQ and the CP2201-GM from Silabs [114]. These controllers could be used for single VTPE-hBEB implementation due to their interrupt support in 8-bit mode.

In order to overcome the drawback of the CS8900A-CQ Ethernet controller, a VTPE-hBEB node composed of two sub-nodes was developed. Each sub-node contains a Microchip PIC18F458 MCU [113] and a Cirrus Logic CS8900A-CQ Ethernet controller (Figure 4.7). This option solves the CS8900A-CQ interrupt problem, increases the overall processing power (due to the use of two MCUs) and allows separating the VTPE-hBEB protocol from the user application. Therefore, one of the sub-nodes runs the VTPE-hBEB protocol while the other runs the user application.

Figures 4.7 and 4.8 illustrate the VTPE-hBEB implementation, which is based on two PIC18F458 MCU/Nicki boards. The Nicki board [115] integrates a CS8900A-CQ controller, a 20MHz crystal, some power supply bypass capacitors and a few resistors, designed to be integrated with the microcontroller.

The MCU drives the interface signals to enable, read, write and reset the CS8900A-CQ controller. Observe that the CS8900A-CQ does not signal the reception of Ethernet frames by means of an interrupt to its host (MCU). Instead, to detect an Ethernet frame reception, a polling mechanism must be used. If the application in the PIC2 has a ready frame to be transmitted

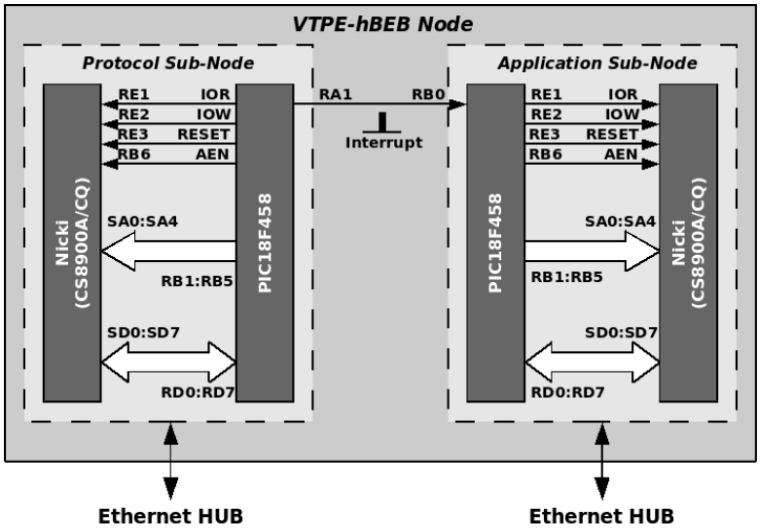

Figure 4.7: Hardware of node based on dual Ethernet controllers.

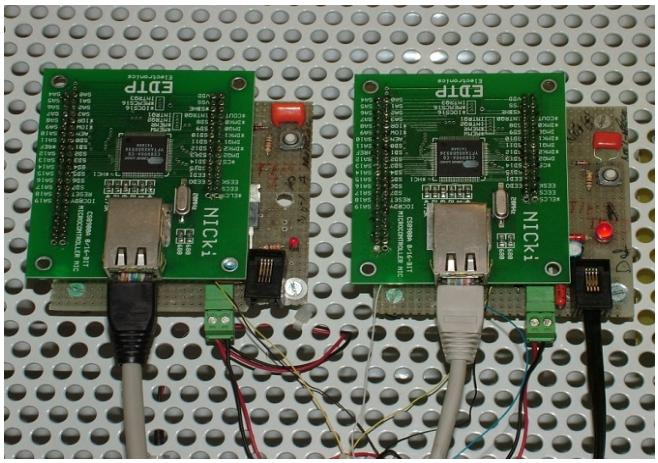

Figure 4.8: Implemented VTPE-hBEB node.

its transmission starts as soon as possible. Conversely, if any transmission is detected during t_{guard}, the ACo counter is also incremented in all VTPE-hBEB stations and, the virtual token is passed to the next node of the virtual ring. Therefore, each VTPE-hBEB station increments its ACo counter whenever a frame is transmitted or just after each idle period equal to t_{guard} followed by an initial idle period equal to $InterFrameGap$.

The Ethernet controller belonging to the protocol sub-node must be programmed in promiscuous mode. This occurs because, in order to separate VTPE-hBEB from standard Ethernet frames, all of them must be accepted

by the controller. On the other hand, due to user requirements, the application (sub-node) Ethernet controller may use any type of addressing (unicast, multicast or broadcast).

4.5 Measurements

The target of the measurements presented in this chapter is to verify how the VTPE-hBEB mechanism is able to cope with the requirements of real-time industrial communications, in the presence of external traffic sources that are out of the sphere-of-control of the RT architecture. Basically, it is assessed the behavior of the VTPE-hBEB mechanism *vs.* the Ethernet standard mechanism, when supporting real-time traffic in a communication medium shared with timing unconstrained traffic.

The measurement metrics analyzed in this chapter include: average delay and token rotation time. The *average delay* is the average delay required to transfer a packet, measured from the start of its generation at the application layer to the end of the packet transfer. The *token rotation time* represents the time interval between two consecutive token arrivals at the same station. This is a very important parameter, as it bounds the shortest update time of periodic variables from the industrial plant.

4.5.1 Measurement setup

Figure 4.9 depicts the general measuring setup, where an Ethernet repeater hub is connected to 6 nodes in a star topology. On the right side of the Figure, 3 RT nodes implement either the VTPE-hBEB mechanism or the standard Ethernet mechanism. Each RT node transfer 1 frame (72 bytes) each $10ms$ at 10 Mbps, whereas each ST station (illustrated on the left) are personal computers running the Distributed Internet Traffic Generator (D-ITG) [116]. This traffic generator was configured to produce UDP packets with constant maximum length (1538 bytes, including IFG, preamble and SFD). A standard (ST) station running the D-ITG generator is capable of producing network loads ranging from 0% to 100% of the network bandwidth (10Mbit). These values can be obtained by increasing or decreasing the inter-departure packet rate (Poisson distributed).

The preliminary experimental results presented in this thesis illustrates a scenario where each ST station imposes an offered load ranging from 30% to

100% of the total network load. Therefore, the total network load imposed by the external stations ranges from 90% to 300% of the total network load (10 Mbps).

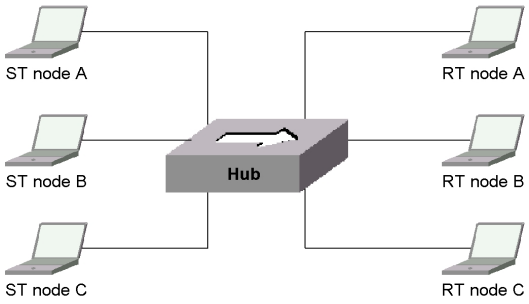

Figure 4.9: The general measuring setup.

The Delay Measurement System (DMS) depicted in Figure 4.10 was built to assess the VTPE-hBEB protocol timeliness. The DMS is composed of a Microchip DSPIC30F6012A microcontroller with appropriate RS232 level converters, among other components.

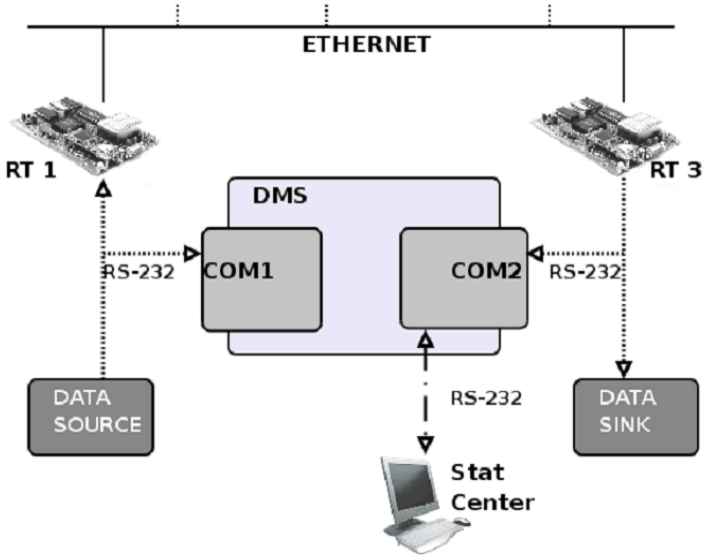

Figure 4.10: Delay Measurement System.

The DMS built-in serial ports are used for byte monitoring, allowing registering the instants in which bytes are transmitted by the data source or received at the data sink. Therefore, it is possible to measure the latency that a byte experiences in the Ethernet bus, as well as its variation and loss.

Following, the DMS is able to compute several variables, namely Average, Minimum and Maximum Delay, Average, Minimum and Maximum Token Rotation Time (TRT), and the Delay and TRT Histogram.

4.5.2 Results

This section presents a preliminary evaluation of the VTPE-hBEB setup. In this sense a test-bed similar to the arrangement showed in Figure 4.9 was used together with the Delay Measurement System showed in Figure 4.10.

The average packet delay for transferring a packet from a RT station is represented in Figure 4.11. These results show that the RT stations implementing the VTPE-hBEB mechanism have a very small average packet delay and, the maximum obtained delay was $5.77ms$. On the other hand, it is also clear that, the average packet delay exponentially increases for ST stations supporting RT traffic, whenever the network load imposed by external (ST) stations just increases from 90% to 120%.

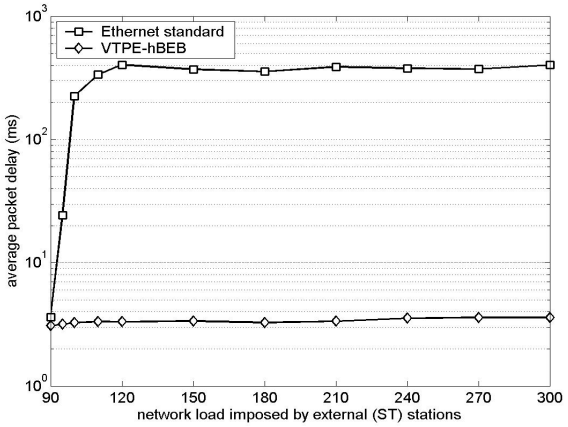

Figure 4.11: Average packet delay.

These results clearly show that, whatever the network load, the average packet delay is nearly constant in the VTPE-hBEB mechanism. These are important results, as they forecast predictable communication delays for the VTPE-hBEB stations for a considerable load range, which is a fundamental requirement to support real-time communication in unconstrained environments. Conversely, as expected, the standard Ethernet stations are not able to provide any RT guarantee when supporting RT traffic.

Another important timing parameter that must be also carefully evaluated is the token rotation time, as the ring stability cannot be affected by external traffic sources with unconstrained timing behavior. We have assessed the behavior of the token rotation time when supporting real-time messages stream with periods of $10ms$.

Figure 4.12 illustrates the impact of external unconstrained traffic upon the average token rotation time in the 90% to 300% load region. In this case, each ST station transmits a variable number of frames with maximum length in order to impose an offered load ranging from 30% to 100% of the total bandwidth.

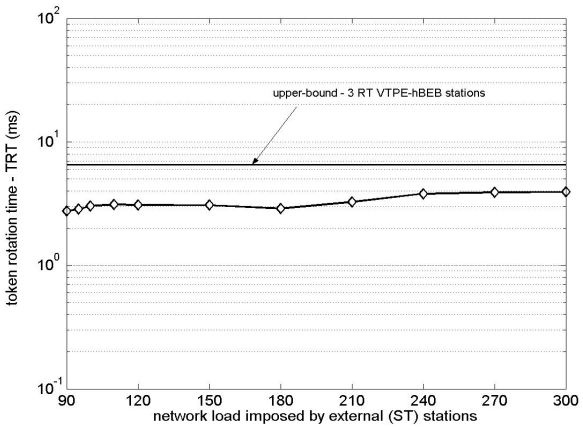

Figure 4.12: Token rotation time.

Figure 4.12 shows that, the token rotation time have a small and almost constant value whatever the offered load imposed by external stations (ST stations). In Figure 4.12 is also plotted the upper-bound for the token rotation time obtained from Equation 4.3. It must be considered that this upper-bound addresses a rarely occurring case, as it is based on the assumption that during a token rotation cycle, none of the VTPE-hBEB station had RT messages to transfer, and that 15 consecutive collisions occurred in the first available slot and; one ST station transferred a message with the maximum allowed length. Most probably, a similar situation will only occur when the AC counters are not synchronized. In such a case, there will be 15 consecutive collisions in the first available slot caused by the VTPE-hBEB stations with inconsistent ACs. And, two or more stations will discard a RT message before reset their AC counters (reset mechanism).

4.6 Summary

The major motivation of this chapter was to propose a communication architecture enabling the support of real-time communications in shared Ethernet environments, where unconstrained Ethernet standard devices (non real-time) coexist with real-time communication devices. To address this problem, it has been proposed the VTPE-hBEB mechanism. This mechanism imposes a higher priority for the transfer of VTPE-hBEB related traffic, guaranteeing the required traffic separation. The analysis included in this chapter shows that, for a moderate number of nodes, a token rotation time of the order of a few milliseconds can be obtained. This figure seems adequate for real-time applications in the automation domain. Class 2 communication requirements can be easily reached using the VTPE-hBEB mechanism and, with some effort, it would possible to reach class 3 applications.

Chapter 5

Understanding the Limitations of the IEEE 802.11e EDCA mechanism

In this chapter, it is analyzed the timing behavior of the IEEE 802.11e protocol, when its EDCA mechanism is used to support real-time traffic. The target of this chapter is to highlight the limitations of the EDCA mechanism when supporting RT traffic in open communication environments. Basically, the timing behavior of the voice category is assessed, when it used to transfer small sized packets generated in periodic intervals. A special emphasis is given to a communication scenario relevant for the next generation communication environments. That is, a wireless error-prone channel, where a set of RT stations is sharing the same frequency band with a variable number of timing unconstrained ST stations. This chapter is largely drawn from the following published work: "Simulation analysis of the IEEE 802.11e EDCA protocol for an industrially-relevant real-time communication scenario" (Moraes et al. [22]).

5.1 Introduction

It is well known that wireless transmissions use error-prone communication channels and have time-variable error characteristics. As a consequence, the use of wireless communication may not be well suited to support RT

communications with high reliability degree [9]. Nevertheless, over the past few years, there has been a growth in the use of wireless technologies in application domains that require a trustworthy Quality of Service (QoS). Therefore, it is reasonable to expect that in the near future, the IEEE 802.11 protocol will spread as a *de facto* standard to support RT communications, mainly due to its simplicity and its high speed *vs.* low cost characteristics. It is worth mentioning that, in today's industrial environments, Ethernet (that have a similar MAC protocol) is the *de facto* communication standard for the lower levels of the industrial communication hierarchy, due to the same positive aspects. This topic was extensively discussed in the previous chapters.

Presently, most of the analyzed solutions to support RT communication in wireless environments address either the Zigbee or the Bluetooth protocols, or some of the wireless extension of the Profibus protocol. It is well-known that IEEE 802.11 protocol is widely used solution. In spite of its relevance, there are few research works addressing specifically the use of this protocol to support RT communications. Besides, traditional performance analysis of the IEEE 802.11 protocol are carried out considering typical multimedia traffic requirements. That is, requirements usually applied for transferring voice and video streams together with background traffic. However, when the communication services are used to support RT applications, specific communication requirements must also be considered, including additional hard real-time and reliability constraints [9].

In this chapter, the new EDCA (Enhanced Distributed Channel Access) mechanism of the IEEE 802.11e amendment is assessed by simulation, in order to understand its behavior when supporting RT communication services. Basically, it is assessed the impact of the timing unconstrained traffic upon the behavior of the voice category, when this access category (AC) is used to transfer small sized packets, generated in periodic intervals. The performed assessment considers an error-prone channel with interferences (*e.g.* Electromagnetic Interference).

This chapter is organized as follows: In Section 5.2 we briefly refer the used SPN simulation model. Afterwards, in Section 5.3 we present the simulation scenarios in detail. In Section 5.4 we carefully discuss the results of the simulations, in order to understand the limitations of the EDCA mechanism when it is used to support RT communication. Finally, this chapter is concluded in section 5.5.

5.2 The Simulation Model

In order to understand the limitations of the EDCA mechanism, it was implemented a *Stochastic Petri Net* (SPN) simulation model that describes the dynamics of the Contention-Based Channel Access function (EDCA) of the IEEE 802.11e standard. This model comprises a precise and detailed implementation of the EDCA function associated to Quality of Service (QoS) stations, considering both their functional and temporal perspectives. Among several implemented functionalities, the following ones can be distinguished:

- Accurate implementation of backoff procedures;
- Frame retransmissions;
- Timeouts and Extended Interframe Spaces (EIFS) modeled according to the standard specifications;
- Transmission opportunities (TXOPs);
- Communication errors, either due to collisions or to external interferences typical of industrial environments (*e.g.* EMI).

Besides, and from a modeling point of view, the model also exhibits an important flexibility in the following aspects:

- Easiness to include modifications or refinements. The model was built in a modular way being composed by several modules, each one implementing a specific functionality. Therefore, their topology helps to localize the modules where modifications/refinements should be included;
- A large number of performance measures of different types can be obtained from the same model without any structural modification;
- The model can be used as a base structure to build more complex and higher-level models (*e.g.* new protocols over the IEEE 802.11e EDCA).

For the assessment of the RT characteristics of the EDCA function it is assumed that, in an infrastructure network, all stations are always in the range of the Access Point (AP); conversely, for *ad hoc* networks, the

same assumption specifies that stations can always hear each other. Thus, communication errors can occur either due to collisions or to external interferences. Therefore, both the hidden and exposed scenarios [9] are explicitly excluded. The complete description of the model implementation (including its validation) is presented in Appendix A.

For all the simulations, it has been used a variant of the Gilbert-Elliot error model, where the channel is always in one of two states: Good or Bad. This model assume that bit errors are independent, with a fixed error rate in each state and, the state sojourn time is log-normal distributed.

According to Willig [117] this model is realistic for wireless transmission in an industrial environment. Then, for the parametrization of the used error model, it has been used the same values as defined in [117], *i.e.*, for all channels the mean duration of good state is $65ms$, the mean duration of bad state is $10ms$ and, the coefficient of variation (CoV) for the bad state holding times has been set to 10 and for the good state to 20. It is assumed that errors occur only in the bad state. Two sets of simulations are performed, differing in their respective mean bit error rate (BER). The first (*error-prone scenario*) set defines a mean BER of 10^{-4}, while the second (*error-free scenario*) set defines that no bit errors occur. Thus, during the bad channel states for the first set, the BER is about 0.00075 and, for the good state no bit errors will occur.

These mean burst lengths lead to a rather bad channel, where the steady-state probability for finding the channel in bad state is approximately 13.3%.

5.3 Simulation Scenario

The simulation scenarios analyze the behavior of the highest access category of the EDCA mechanism (*voice*), when this category is used to transmit real-time data (small sized packets generated in periodic intervals) from RT stations, in the presence of unconstrained traffic sources, that are out of the sphere-of-control of the RT architecture. These RT periodic data exchanges are intended to model both sensor messages sent to plant controllers, and output messages sent from plant controllers to the actuators.

Therefore, a simulation model was built consisting in an *ad hoc* network topology, where multiple ST and RT stations operate in the same frequency band (Figure 5.1). The RT stations only transfer RT traffic, using the default set of parameters defined by the EDCA function for the voice (VO)

access category. On the other hand, ST stations (unconstrained stations) transmit three types of traffic: voice (VO), video (VI) and background (BK) traffic, also using the default set of parameters defined by the EDCA function for these categories.

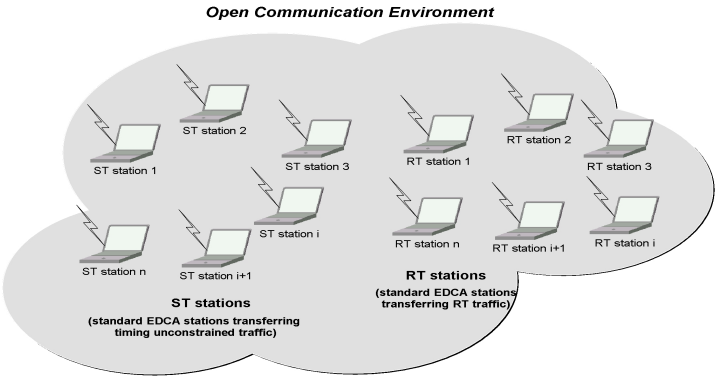

Figure 5.1: Simulation scenario.

Basically two simulation cases are analyzed. The first scenario (*small population case*) considers 10 ST stations operating in the same frequency band together with 10 to 40 RT stations. The second scenario (*large population case*) extends the number of ST stations to 40. The physical parameters used in the simulations are based on the IEEE 802.11a PHY mode [118], which are summarized in Table 5.1. Specifically, each station operates at OFDM (Orthogonal Frequency Division Multiplexing) PHY mode, control frames are transmitted at a basic rate equal to 1 Mbps, while the MSDU (MAC service data units) are transmitted at 36 Mbps.

Table 5.1: Simulation parameters for MAC and 802.11a PHY layers.

Parameters	**Value**
aSIFSTime	16 μs
aSlotTime	9 μs
aCCATime	4 μs
aAirPropagationTime	1 μs
aRxTxTurnaroundTime	2 μs
aPreambleLenght	20 μs
aPLCPHeaderLenght	4 μs

The RT traffic is characterized by periodic sources with a small amount of imposed jitter. To model this behavior a normal distribution with $\sigma/\mu \leq 1\%$ (σ is the standard deviation and μ is the average expected value) is used

to generate the periodic traffic sent by RT stations. It is also guaranteed that the RT traffic is not correlated among RT stations. The ST stations have Poisson traffic sources. The maximum number of transmission attempts is set to 4. The MAC queue size is set to 50 positions. All other relevant simulation parameters are shown in Table 5.2.

Table 5.2: Simulation data.

Parameters	**RT stations**	**ST stations**		
		VO	VI	BK
CW_{min}	7	7	15	31
CW_{max}	15	15	31	1023
AIFSN	2	2	3	7
TXOP (ms)	1.504	1.504	3.008	0
Packet Size - bytes	45	160	1280	1600
Message stream (ms)	2, 10, 20	Variable	Variable	Variable

The generated data frames have a constant size. Each RT station generates packets with fixed message stream periods (MSP) of $2ms$, $10ms$ or $20ms$, with 45 bytes for data payload. This is equivalent to generate 500, 100 or 50 packets/s. Therefore, each RT station imposes a constant network load of 180, 36 or 18 kbits/s that represents less than 0.5%, 0.1% or 0.05% of the total network load (without considering the MAC and PHY headers). That is, the overall real-time traffic represents less than about 20% of the total network load (for the case of 40 RT stations and $MSP = 2ms$).

For the set of ST stations, the offered load (denoted as G_{ST}) ranges from 10% to 90%. Each ST station generates λ voice, video and background packets/s at the same rate, in order to impose the requested G_{ST} overall network load. The arrival rate (λ) can be obtained by:

$$\lambda = \frac{G_{ST}}{(PK_{VO} + PK_{VI} + PK_{BK})} = (packets/s) \tag{5.1}$$

where G_{ST} ranges according to the requested percentage (10% to 90%) of the PHY data rate (36 Mbps) and, PK_{VO}, PK_{VI} and PK_{BK} represent the packet size for data payload (bits) transmitted in each access category by the ST stations. The obtained value for the arrival rate λ is equally divided among all the ST stations. Table 5.3 summarizes the number of VO, VI and BK packets generated by each ST station, in order to impose the desired offered load (10% to 90%).

Table 5.3: Number of packets/s generated by ST stations.

Network load	**ST stations**	
	Small population Packets/s	Large population Packets/s
10%	14.8	3.7
20%	29.6	7.4
30%	44.4	11.1
40%	59.2	14.8
50%	74.0	18.5
60%	88.8	22.2
70%	103.6	25.9
80%	118.4	29.6
90%	133.2	33.3

5.4 Simulation Results

All the simulation results have been obtained using the SPN simulation model (that is described in Appendix A), with 95% confidence interval and a half-width interval of 5%. The performance metrics analyzed in this chapter include: throughput, packet loss, average delay and average queue size. The *throughput* is the ratio between the total number of successfully transferred packets and the total number of generated packets for each traffic stream. Therefore, it represents the relative throughput. The *packet loss* metric is computed as $(1-throughput)*100$ and represents the percentage of packets that are lost for each traffic stream. The *average delay* is the average delay required to transfer a packet, measured from the start of its generation at the application layer to the end of the packet transfer. The *average queue size* represents the average output buffer occupancy.

5.4.1 Simulation Results: RT Traffic

As a first step, a set of simulations was performed to characterize the network behavior, when just the RT stations are transferring messages in the communication medium. Thus, 10 to 40 RT stations generating RT packets (45 bytes for data payload) with fixed MSP of $2ms$, $10ms$ or $20ms$ were considered. This simulation scenario describes the network behavior when there are no ST stations trying to transfer its own messages, *i.e.*, it represents an unrealistic *closed* communication environment. Therefore, the simulation results obtained in this section are intended just for compari-

son purposes. In subsection 5.4.2, it is assessed the impact of ST traffic (perturbation effect) upon the transfer of RT traffic.

The obtained values for the average queue size, relative throughput and the average packet delay are shown in Figures 5.2, 5.3 and 5.4, respectively. From such results, it is clear that, the EDCA mechanism is not able to provide a RT communication service for more than 20 RT stations generating packets with MSP of $10ms$ or $20ms$. This conclusion derives from the following observations: on the one hand, the average queue size is smaller than 1 packet, which is an indication that, in average, all the deadlines for the RT traffic are accomplished (considering that, in the most usual case, the deadline of a message stream is equal to its generation period) and; the average packet delay is smaller than the period of the related message stream ($10ms$ or $20ms$). However, on the other hand, the relative throughput is smaller than 0.9 for a number of stations larger than 20. This means that the RT message streams have lost more than 10% of its deadlines and, despite many real-time applications being loss-tolerant, this value cannot be well accepted by some applications. Loss tolerance is different for different type of applications *e.g.*, a video stream may be specified to tolerate a maximum of 10% deadline loss rate, only when the lost frames are "adequately" spaced. Similar values have also been pointed out for networked control systems scheduled according to the (m,k)-firm model [12].

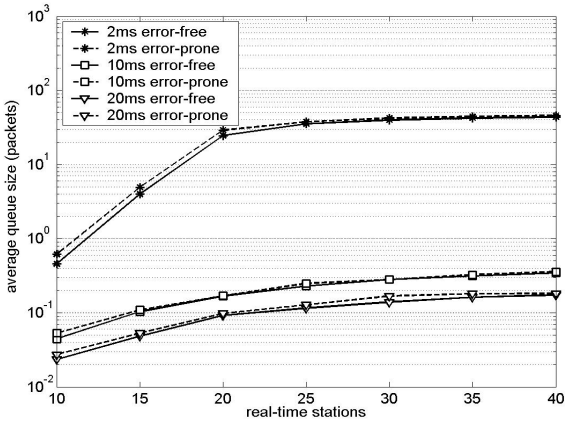

Figure 5.2: Average queue size: undisturbed scenario.

Applying a similar reasoning, it can be also observed that for RT message streams with $MSP = 2ms$, the EDCA mechanism is not able to provide an acceptable real-time service when the number of RT stations exceed 10

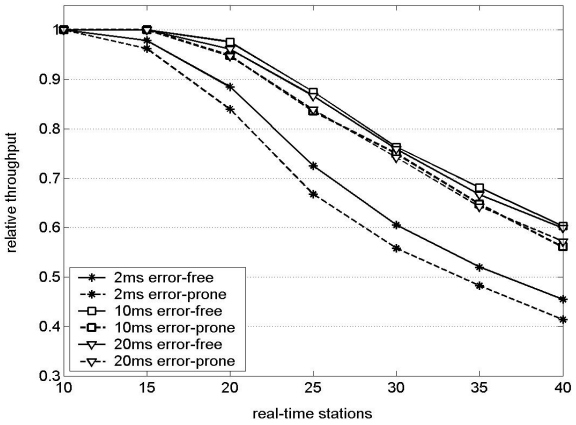

Figure 5.3: Throughput: undisturbed scenario.

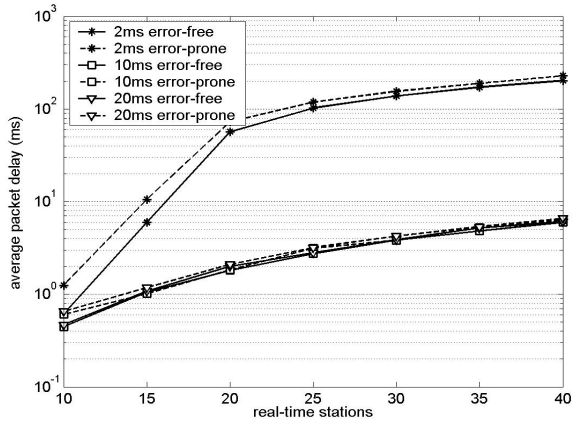

Figure 5.4: Average packet delay: undisturbed scenario.

stations. It can be easily verified through the average queue size results (Figure 5.2) that the average number of packets in the queue is already approaching 1 packet even for a number of RT stations as low as 10 stations.

Therefore, from the analysis of this section, it can be concluded that the the relevant scenarios for the assessment of the RT characteristics of the EDCA mechanism are restricted to 10/20 RT stations generating packets with MSP of $10ms$ or $20ms$ and; 10 RT stations generating packets with MSP of $2ms$. Simulation scenarios above these thresholds are no longer relevant, as the EDCA mechanism is not able to support a RT communication service even for the case of an undisturbed scenario.

5.4.2 Simulation Results: ST-RT Traffics

In this section, it is assessed the behavior of both the small and the large population scenarios, when ST stations are joined with RT stations, both transferring its own messages in the same wireless domain (operating in the same frequency band). For the sake of simplicity, only the values for RT traffic are plotted in the following figures, as the target of this study is to illustrate the impact of the external timing unconstrained traffic upon the real-time traffic, for each of the real-time configuration (10 RT stations with a fixed MSP of $2ms$; 10/20 RT stations with a fixed MSP of $10ms$ or $20ms$). Furthermore, Ni *et al.* [93, 119] have already demonstrated that the EDCA mechanism improves the performance behavior for high priority traffic by downgrading the service of the low-priority one. Thus, it is not relevant to present the behavior of the ST traffic.

5.4.3 The impact of timing unconstrained ST traffic upon the average queue size of RT stations

A first simulation analysis concerns the assessment of the average queue size in a RT station, when the RT traffic in the wireless domain is disturbed by the presence of timing unconstrained traffic from ST stations. Figure 5.5 shows the average queue size for message stream periods of $2ms$, considering the case of 10 RT stations operating in both the small and the large population scenarios (10ST - 40ST).

When comparing the undisturbed scenario with the case where 10 RT stations operating together with 10/40 ST stations ($MSP = 2ms$), it becomes clear the impact that the ST timing unconstrained traffic has upon the average packet size of real-time messages. Figure 5.5 also illustrates the impact of the increasing number of stations contending for the medium access, which shows a clear degradation of the Quality of Service for the large population scenarios. Summing up, the EDCA mechanism is not able to provide RT communication service for MSP of $2ms$ in open communication environments, since that the average queue size is already larger than 1 packet when the network load imposed by external stations is only slightly above 10%.

Another important result, that is in contradiction with the common belief of wireless communications, is that the major source of perturbation upon the RT communication is caused by the external network load and not

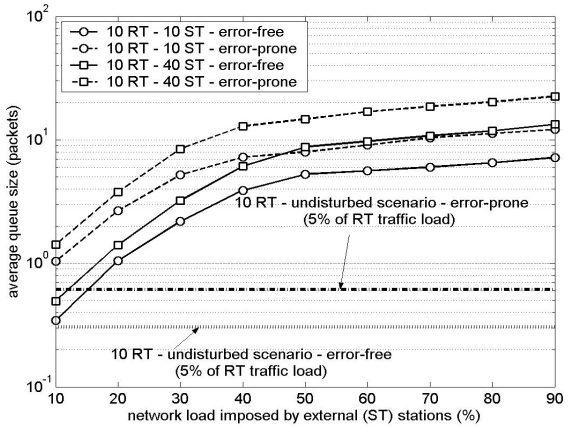

Figure 5.5: Average queue size (small and large pop.): error-free *vs.* error-prone - $MSP = 2ms$.

by the error-prone characteristics of the wireless medium. It is interesting to note that the average queue size increases more than one order of magnitude when the network load increases from 10% to 50%, whereas the average queue size for an error-prone channel is only slightly larger that the average queue size for an error-free channel.

Consider now the case of RT stations with MSP of $10ms$ operating in both the small and large population scenarios. Consider also that the transfer of RT messages is disturbed by timing unconstrained ST traffic sources. Figure 5.6 illustrates the average queue size for the case of 20 RT stations. From Figure 5.6, it can be observed the impact of the timing unconstrained ST traffic upon the number of packets waiting to be transmitted. Whatever the external traffic load, the average number of queued packets is always kept under 1 packet. This indicates that the EDCA mechanism can be suitable to support real-time traffic, when the message stream periods are of $10ms$. Similar results were obtained for MSP of $20ms$, and for the case of 10 RT stations sending messages with MSP of $10ms$ or $20ms$. Therefore, those results can be generalized, stating that, in what concerns the average queue size of RT messages, the EDCA mechanism can be suitable to support real-time traffic in disturbed communication environments.

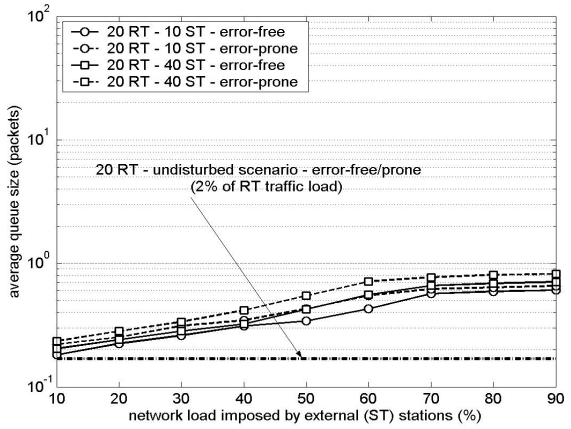

Figure 5.6: Average queue size (small and large pop.): error-free *vs.* error-prone - $MSP = 10ms$.

5.4.4 The impact of timing unconstrained ST traffic upon the throughput of RT stations

A second simulation analysis concerns the assessment of the relative throughput that can be handled by RT stations, when the communication environment is disturbed by timing unconstrained ST traffic. Figures 5.7 and 5.8 show the throughput results for message stream periods of $10ms$ and $20ms$, in both the small and large population scenarios. The other plots representing the percentage of packet loss can be inferred from these Figures, as the percentage of packet loss is given by $(1 - throughput) * 100$.

Figures 5.7 and 5.8 show that, for a number of RT stations above 10, the impact of the timing unconstrained ST traffic upon the throughput of real-time messages becomes clearly undesirable. It is clear that, the EDCA mechanism is not able to provide any acceptable RT guarantee for a number of RT stations above 10 stations, as the relative throughput is smaller than 0.9 (the percentage of packet losses is above 10%), when the network load imposed by external stations increases just from 10% to 30%. It is well known that real-time applications can accommodate some packet loss without noticeable degradation in the quality of service *e.g.*, for voice, depending on the encoding and transmission schemes used, more than 10% packet loss can still be acceptable [120], but, in general, it is already clearly undesirable even for typical soft real-time applications.

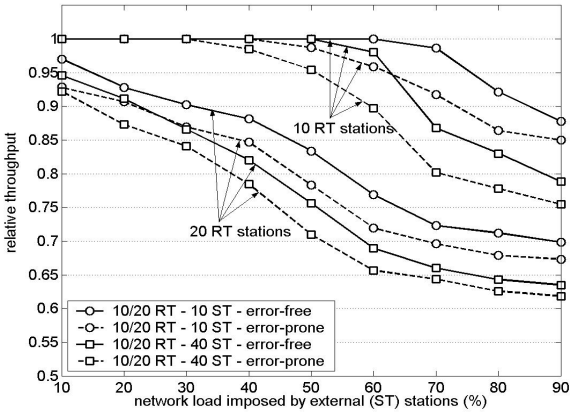

Figure 5.7: Throughput (small and large pop.): error-free *vs.* error-prone - $MSP = 10ms$.

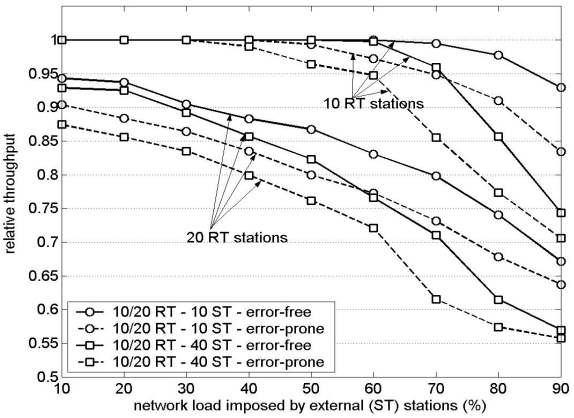

Figure 5.8: Throughput (small and large pop.): error-free *vs.* error-prone - $MSP = 20ms$.

5.4.5 The impact of timing unconstrained ST traffic upon the average packet delay of RT stations

Finally, the average packet delay for transferring a packet in small and large population scenarios is assessed. As concluded through the average queue size and throughput results, the EDCA mechanism is not able to provide a real-time communication service for more than 10 RT stations, even for message stream periods of $10ms$ or $20ms$. Then, the average packet delay will be plotted only for the case of 10 RT stations with MSPs of $10ms$ and

$20ms$, that are represented in Figures 5.9 and 5.10, respectively.

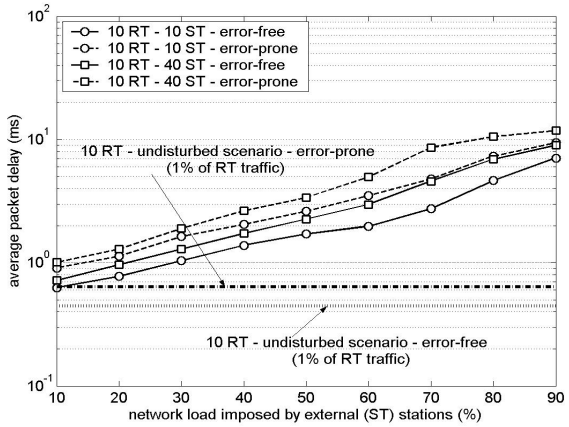

Figure 5.9: Average delay (small and large pop.): error-free *vs.* error-prone - $MSP = 10ms$.

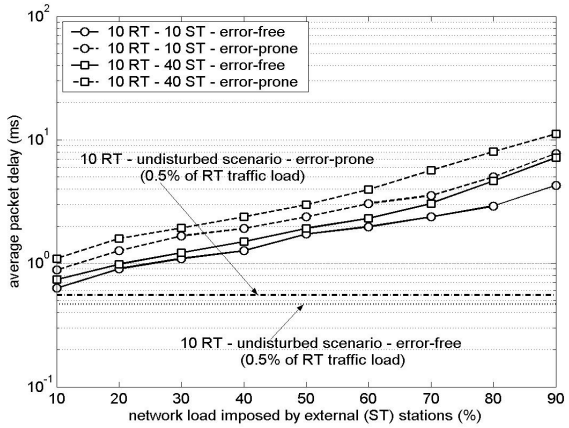

Figure 5.10: Average delay (small and large pop.): error-free *vs.* error-prone - $MSP = 20ms$.

From this set of simulations (Figures 5.9 and 5.10), it can be concluded that the EDCA mechanism is able to support the RT traffic with MSPs of $10ms$ or $20ms$ generated by up to 10 RT stations, even for the case where such traffic is subject to external disturbances of up to 75% of external network load. Additionally, it can be also observed the higher impact of traffic generated by ST stations upon the average packet delay of RT messages. Secondly, and most interestingly, when the network load increases, the effect of the error-prone channel becomes more relevant. However, when

comparing the relative throughput in error-prone *vs.* error-free channels, it can be observed that the difference is always close to 5%. Conversely, this difference is much larger when comparing the increase of the network load imposed by external ST stations.

5.5 Summary

In this chapter, the suitability of the IEEE 802.11e EDCA protocol to support real-time communication scenarios in open communication environments has been assessed. The simulation scenarios consider the existence of an external network load (timing unconstrained traffic generated by generic ST stations) and an error-prone channel. Their impact upon the transfer of real-time traffic is assessed. Basically, we assessed the behavior of the highest access category of the EDCA mechanism (voice), when this access category is used to transfer small sized packets generated at periodic intervals. The simulation analysis shows that:

- The increasing number of stations in the network domain strongly influences the RT traffic behavior, by means of higher average packet delays, higher percentage of packet loss, higher average queue size and smaller throughput;

- The average queue size increases almost one order of magnitude when the network load increases from about 10% to 30% for message stream periods of $2ms$. It becomes clear the undesirable effect that the timing unconstrained traffic generated by ST stations has upon the transfer of real-time messages. These are very important results, as they forecast unacceptable communication delays when supporting real-time communications in shared communication environments;

- The EDCA function is not able to support RT message streams with periods of $2ms$, and it is able to support *at maximum* 10 RT stations when transmitting RT message streams with periods of $10ms$ or $20ms$.

- The major impact upon the quality of the RT communication is due to collisions, imposed by the timing unconstrained traffic. The error-prone characteristics of wireless medium have a significantly smaller impact.

Therefore, the main conclusion of the simulated scenarios is that the default parameter values of the EDCA mechanism are not able to guarantee the real-time communication requirements, when the voice priority is used to support real-time traffic in shared communication environments, unless the number of RT stations is kept below 10 stations, and the MSPs are kept as high as $10ms$. Therefore, new communication approaches must be devised in order to enable the use of IEEE 802.11e networks to support real-time communication in the next generation communication environments.

Chapter 6

The VTP-CSMA architecture

In this chapter, a new wireless architecture is proposed intended to provide RT communication in IEEE 802.11e standard networks. The VTP-CSMA architecture considers an unified wireless system in one frequency band, where the communication bandwidth is shared by real-time and non real-time communicating devices. It is based on a Virtual Token Passing procedure (VTP) that circulates a virtual token among real-time devices. This virtual token is complemented by the underlying traffic separation mechanism (TSm) described in chapter 3. This chapter is largely drawn from the following published works: "Real-Time Communication in 802.11 Networks: Timing Analysis and a Ring Management Scheme for the VTP-CSMA Architecture" (Moraes et al. [21]); "VTP-CSMA: A Virtual Token Passing Approach for Real-Time Communication in IEEE 802.11 Wireless Networks" (Moraes et al. [19]); "Real-Time Communication in 802.11 Networks: The Virtual Token Passing VTP-CSMA Approach" (Moraes et al. [20]).

6.1 Introduction

The IEEE 802.11 architecture provides a wireless LAN that supports station mobility transparently to the upper layers. The basic service set (BSS) is the building block of an IEEE 802.11 WLAN, which actually provides two types of configurations: *independent BSS (IBSS)* and *infrastructure*. The IBSS is the most basic type for a IEEE 802.11 WLAN, which may be composed of, at minimum, two stations. This mode of operation is often referred to as *ad*

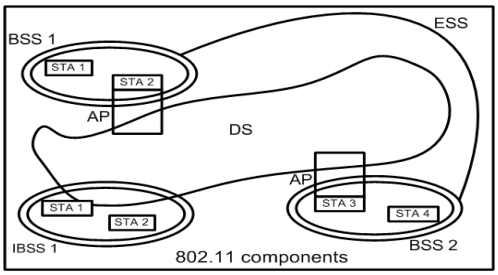

Figure 6.1: Extended service set network.

hoc. The infrastructure mode includes one or more access points (AP) that convey the communication among wireless stations. It is possible to create a wireless network of arbitrary size and complexity, where several BSSs may be interconnected, appearing as a single BSS at the Logical Link Control (LLC) layer [89]. The IEEE 802.11 standard refers to this type of network as the *Extended Service Set* (ESS) network (Figure 6.1).

To become a member of an infrastructure BSS, a station needs to be *associated*. These associations are dynamic; stations within an ESS may communicate amongst each other and mobile stations may transparently move from one BSS to another. As a consequence, all of the following operating modes are possible:

- the BSSs may partially overlap;
- the BSSs can be physically disjointed; the BSSs may be physically collocated;
- one (or more) IBSS or ESS networks may be physically present in the same space as one (or more) ESS network(s).

Therefore, any IEEE 802.11 environment is a relatively open communication environment. As a consequence, the system load cannot be predicted at system setup time, nor can be effectively controlled during the system run-time.

The VTP-CSMA architecture has been proposed to deal with this problem. This proposal considers an unified IEEE 802.11/802.11e wireless system operating in one frequency band, where the bandwidth is shared by all the communicating devices. The main goal of the VTP-CSMA architecture is to overcome the limitations of the EDCA function (discussed in chapter 5) to support RT communication.

The remainder of this chapter is organized as follows. In section 6.2, the VTP-CSMA proposal is presented in detail. A worst-case timing analysis is presented in section 6.3 demonstrating that the token rotation timing is upper-bounded, even in the presence of timing unconstrained traffic. Therefore, it shows that the VTP-CSMA architecture is able to provide a RT communication service, when the communication medium is shared with timing unconstrained stations (ST stations). In sections 6.4 and 6.5, the timing behavior of the VTP-CSMA architecture is analyzed, both for average case (by simulation) and worst-case scenarios. In Section 6.6 a ring management procedure is presented, which allows stations to dynamically leave or join (rejoin) the VTP-CSMA architecture. This procedure enables the support of highly dynamic communication scenarios with multiple devices opening/closing RT connections, such as those found in VoIP (voice over IP) applications. Finally, some conclusions are drawn.

6.2 The VTP-CSMA architecture

The VTP-CSMA architecture is intended to provide a real-time communication service in IEEE 802.11/802.11e networks. This architecture allows the coexistence of default (non real-time) stations with enhanced (real-time) stations in the same network domain, imposing higher priority to real-time traffic. This means that it becomes possible to support the real-time communications generated by a subset of network stations, without the need to upgrade the firmware of all the communicating devices. One of the main advantages of the VTP-CSMA architecture is that only the NICs firmware of the real-time stations must be updated according to the VTP-CSMA proposal (compatibility *level 3*)[1]. Besides, the non real-time stations can be standard IEEE 802.11/802.11e devices, which may initiate any communication at any time, *i.e.* the non real-time stations are completely out of the sphere-of-control of the RT architecture.

The VTP-CSMA architecture is based on the control of the medium access right, by means of a Virtual Token Passing (VTP) procedure among RT stations (hereafter also referred as a VTP-CSMA station), complemented by a traffic separation mechanism (TSm) which guarantees that, whenever

[1] The compatibility levels was defined in chapter 2. The compatibility level 3 comprises RT communication proposals able to offer RT guarantees in presence of *third* devices, requiring just modifications at the firmware/software level of the RT communicating devices.

a RT station is contending for the medium access, it will win the contention prior to any other ST station. Such underlying traffic separation mechanism was previously described in chapter 3. Therefore, whenever a collision between one RT station and a set of ST stations occurs, all the involved stations, except the RT one, will use the prioritized medium access mechanism (EDCA) and select a random backoff interval according to the access category (voice, video, best-effort and background). Conversely, the RT station transfers its own traffic at the highest access category, using the highest priority level of the EDCA mechanism, *i.e.*, setting the Arbitration Interframe Space (AIFS) to:

$$AIFS[VO] = aSIFSTime + 2 \times aSlotTime$$

and the contention window (CW) to:

$$aCWmin[VO] = aCWmax[VO] = 0$$

This means that any VTP-CSMA station will always try to transmit its frame in the first EDCA available slot (*forcing collision resolution* approach), while all the other ST stations will wait during a time interval evaluated by the local backoff functions. Nevertheless, if two or more VTP-CSMA stations simultaneously contend for the medium access, they would collide and eventually discard the frame (after the maximum number of retransmission attempts). This behavior is overcome by means of a *Virtual Token Passing* (VTP) procedure, which serializes the transmission of the VTP-CSMA stations.

The VTP procedure considers a process group G with np members. Initially, it is just considered a static communication environment, with a fixed number of RT stations. The membership is represented as $L = \{NA_1, NA_2, ..., NA_{np}\}$. The notion NA_i denotes the *i-th* station in G and is also used as station identification (ID) for NA_i. Each station in L maintain a local access counter (ACo), which is used as a basic mechanism to circulate a virtual token in L. The generic *i-th* VTP-CSMA station *captures* the virtual token when ACo equals NA_i. If the station has queued messages, then it will immediately transfer them during a time interval upper bounded by the transmission opportunity period (TXOP). The underlying TSm mechanism guarantees that the VTP-CSMA station will win the medium access contention. At the end of the current TXOP, each VTP-CSMA station will increase its ACo value, passing the virtual token to the

next station (station with NA_i=$ACo+1$)[2]. Whenever the VTP-CSMA station holding the token does not have any RT message to transfer, it will allow default stations to contend for the medium access, during a time interval $t_2 = aSIFSTime + 3 \times aSlotTime$. As the EDCA mechanism allows any station to start a transmission after $aSIFSTime + 2 \times aSlotTime$, it enables the coexistence of RT stations with non-real-time ST stations in the shared communication environment.

Figure 6.2 illustrates the VTP-CSMA mechanism, that is represented by four procedures: *Initialization*, *Main*, *Transmission* and *Listening*. According to the *Initialization* procedure (line 2, Figure 6.2), the access counter (ACo) value is set to NA_1 in all VTP-CSMA stations. Three local variables (t_1, t_2, t_3) of type integer are defined, where t_1 and t_2 are slot time counters, whereas t_3 is a special collision counter used for re-initialization purposes. The *Main* procedure is executed at the beginning of each time slot[3], where every VTP-CSMA station firstly verifies if the special collision counter (t_3)[4] exceeds the maximum number of retransmission attempts (RN). Whenever this number is exceeded, the *Initialization* procedure is executed (reset mechanism). Afterwards, depending on both the channel event during the last slot and the ACo value, each VTP-CSMA station will take a specific action.

The *Transmission* procedure is only executed by the VTP-CSMA station holding the token ($ACo = NA_i$) and, it is only initiated after the medium being idle during $AIFS[VO]$, *i.e.* the minimum AIFS value for the EDCA mode, as defined by IEEE 802.11e. Conversely, the *Listening* procedure may be initiated after the medium being idle during SIFS ($aSIFSTime$).

[2] The ACo must be increased by "mod" operation, *i.e.* $ACo = (ACo \bmod np) + 1$. Therefore, when it is referred that the ACo is increased, it is executed such "mod" operation.

[3] More accurately, the operation performed by the VTP-CSMA architecture shall be determined on specific slot boundaries, which are defined by the Enhanced Distributed Channel Access Function (EDCAF) of the EDCA mechanism. On specific slot boundaries, each station performs one and only one operation. For further details see subsection 9.9.1.3 of the IEEE 802.11e amendment.

[4] When a collision resolution starts, it can be a consequence of three different types of collisions: (1) among ST stations; (2) among ST stations and the active VTP-CSMA station that holds the token; (3) among VTP-CSMA stations. The first scenario can be detected if the communication medium remains idle during the time interval t_2. The second scenario is easily solved in favor of the VTP-CSMA station. Finally, when multiple VTP-CSMA stations simultaneously contend for the medium access, it means that the ACo counters of the multiple VTP-CSMA stations lost their synchronization, forcing the

```
1  ∀ process NA_i
2  Initialization: ACo ← NA_1; t_1 ← 0; t_2← 0; t_3← 0;
3  Main: while t_3 ≤ RN do
4            if ACo = NA_i then
5               {wait AIFS[VO]};
6               NA_i executes Transmission;
7            else
8               {wait SIFS};
9               NA_i executes Listening;
10           endif
11       end while
```

Figure 6.2: VTP-CSMA mechanism.

Five channel states are defined (determined at the beginning of each time slot):

1. *transmission* from other stations: One or more messages are being transmitted over the channel.

2. $successful\_transmission$ from other stations: The channel is idle, and a successful message transmission from other station finished one time slot ago.

3. channel *continuing_idle*: The channel is idle and was also idle one time slot ago.

4. channel $idle\_after\_collision$: The channel is idle, and there was a collision one time slot ago.

5. $successful$ transmission: The channel is idle, "I am the transmitting station, and I finished the transmission of one or more messages (upper bounded by TXOP interval)" one time slot ago.

According to these channel states and the ACo value, each VTP-CSMA station takes a specific action. Firstly, whenever the VTP-CSMA station *captures* the virtual token ($ACo = NA_i$), it will execute the *Transmission* procedure (Figure 6.3). This procedure works as follows. If the VTP-CSMA station holding the token have a RT message to transfer, it will immediately start the transmission. If a collision occurs, the VTP-CSMA

re-initialization of the ring.

station increments its t_3 counter and, it will retry the transmission until the maximum defined number of transmission attempts (RN). Whenever, a successful transmission occurs, the VTP-CSMA station holding the token will execute the *Listening* procedure (lines 4-5, Figure 6.3), where each VTP-CSMA station will increase its ACo value, passing the virtual token to the next station. Conversely, whenever the VTP-CSMA station holding the token does not have any RT message to transfer, it will allow default stations to contend for the medium access, during a time multiple of $aSlotTime$.

```
1  Transmission:
2  if NA_i has messages to be transmitted then
3      start the transmission; wait for transmission to complete;
4      if successful transmission then
5          t1← 1; t3← 0; go to Listening;
6      else
7          t3++;
8          if t3 ≤ RN then
9              go to Transmission;
10         endif
11     endif
12 else
13     t2← 2; go to Listening;
14 endif
```

Figure 6.3: Transmission procedure.

As illustrated in Figure 6.2, all VTP-CSMA stations that do not have the token ($ACo \neq NA_i$) will execute the *Listening* procedure (Figure 6.4), and depending on the channel state, these VTP-CSMA stations take a specific action:

1. *transmission* from other stations (lines 5-7): All VTP-CSMA stations wait for the end of transmission and then update the variables t_1 and t_2;

2. $successful\_transmission$ from other stations (lines 8-9): All VTP-CSMA stations update t_1, t_2 and t_3;

3. channel *continuing_idle* (lines 10-17): All VTP-CSMA stations increment t_2 and, verify the value of t_1. If $t_1 = 1$, all VTP-CSMA stations increment its ACo value. That is, $t_1 = 1$ and the channel *continuing_idle* state means that a successful transmission occurred

and, a TXOP period finished. Besides, t_2 will be incremented and, each time that t_2 value is greater or equal than 3, all VTP-CSMA stations must also increment its ACo value;

4. channel *idle_after_collision* (lines 18-19): All VTP-CSMA stations increment t_2 and t_3.

```
1 Listening:
2 At the beginning of the next slot
3   event ← channel event during the last slot;
4   switch (event)
5       case transmission:
6           wait for transmission to complete;
7           t1 ← 0; t 2← 0; break;
8       case successful_transmission:
9           t1 ← 1; t2 ← 1; t3 ← 0; break;
10      case continuing_idle:
11          t2++;
12          if t1 = 1 then
13              ACo++; t1 ← 0; t2 ← 0; break;
14          endif
15          if t2 ≥ 3 then
16              ACo++; t1← 0; t2← 0; t3← 0; break;
17          endif
18      case idle_after_collision:
19          t2++; t3++; break;
20  end switch
```

Figure 6.4: Listening procedure.

Most likely, when a collision occurs involving just ST stations (EDCA stations), the next events will be followed by *continuing_idle* channel states, due to the selected backoff interval ($CW_{min} = 7$, for voice category). However, when a VTP-CSMA station is involved in the collision, the next channel state will be always a *transmission* ($CW_{min} = CW_{max} = 0$). Therefore, the reset mechanism (*Initialization* procedure) is activated whenever there are RN consecutive collisions without any idle time greater than t_2 ($aSIFSTime + 3 \times aSlotTime$) among collisions.

Figure 6.5 illustrates the behavior of the proposed VTP-CSMA mechanism. Assume that three VTP-CSMA stations, $L = \{NA_1, NA_2, NA_3\}$, are sharing the physical medium with 1 EDCA station (ST station). To simplify the example, it is considered only the voice access category for the ST station ($AIFSN[VO] = 2$). According to the initialization procedure (line

2, Figure 6.2), the ACo value is equal to NA_1 in all VTP-CSMA stations after *aSIFSTime* (instant *a1*). Moreover, as this initial version handles only a fixed number of stations, each VTP-CSMA station knows the total number of RT stations ($np = 3$). Therefore, station NA_1 (token holding station) runs the *Transmission* procedure, while stations NA_2 and NA_3 execute the *Listening* procedure. However, as station NA_1 has no message to be transferred (until instant $a2$), this station will also execute the *Listening* procedure. This procedure consists in continuously monitoring the channel. According to the example, the next channel states are followed by channel *continuing_idle* periods. Therefore, all VTP-CSMA stations ($ACo \neq NA_i$) will start the slot counter incrementing counter t_2 (lines 10-17, Figure 6.4). Then, counter t_2 will expire in all stations at instant *a3* (*3*× *aSlotTime*) and the virtual token is passed to station NA_2 (lines 15-17). As the channel remains idle (NA_2 station has no message to be transmitted), counter t_2 will expire again at instant *a5*, passing the virtual token to station NA_3.

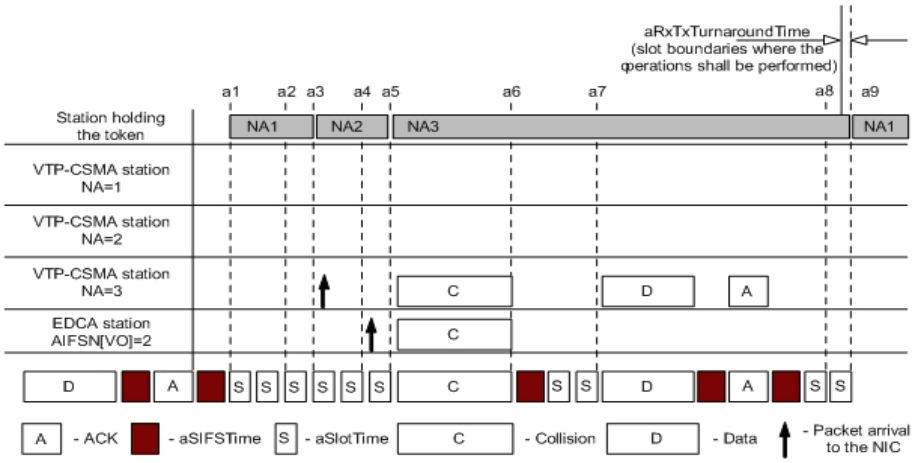

Figure 6.5: Behavior of the VTP-CSMA mechanism.

This station may now start to transfer its own real-time messages. However, considering that between instants *a4* and *a5*, the ST station generated one packet, both stations will start the transmission at the same time. Consequently, a collision will occur and the VTP-CSMA station (NA_3) will be able to detect it by the absence of the ACK frame. According to the underlying TSm mechanism, it will retry the transmission after $aSIFSTime + 2 \times aSlotTime$ following the end of the medium busy condition. This means that NA_3 will start the transfer of its real-time messages at instant *a7* (considering that the ST station selected a backoff value greater than 0). It is worth noting that stations NA_1 and NA_2 after in-

stant *a6* do not detect the $successful\_transmission$ channel state and after $aSIFSTime$ it is detected the channel $continuing\_idle$ state and, the slot counter starts incrementing t_2 (line 11, Figure 6.4). However, the beginning of a transmission is detected (instant *a7*) before the expiration of counter t_2. Therefore, station NA_3 continues the transmission and both NA_1 and NA_2 stations must wait for its completion (line 6, Figure 6.4).

Afterwards, when the end of transmission is detected ($successful\_transmission$ channel state), if there were still packets in the queue and enough TXOP time to transmit another packet, including its response frame, the token holding station NA_3 would start another transmission just after $aSIFSTime$. Therefore, when a station transmits multiple frames, the ACo is incremented only at the end of TXOP. As the station NA_3 has no more packets to be transmitted, all VTP-CSMA stations detect the $successful\_transmission$ channel state and the variable t_1 is incremented (lines 8-9, Figure 6.4). In the next event (channel $continuing\_idle$ state), the virtual token is passed to the next station (instant *a9*) by incrementing the ACo counters in all the VTP-CSMA stations.

Summing up, each VTP-CSMA station will increment its ACo value on specific slot boundaries[5], as follows:

- Following an idle medium duration of ($aSIFSTime + 2 \times aSlotTime - aRxTxTurnaroundTime$) after the last busy medium on the antenna that was the result of a reception of a frame with a correct FCS (Frame Check Sequence);

- Following and idle medium duration of ($aSIFSTime + 3 \times aSlotTime - aRxTxTurnaroundTime$) after the last busy medium on the antenna for all other cases (including the transmission of a frame that did not require an acknowledgement).

[5] The specific slot boundaries at which exactly one of these operations shall be performed are defined in section 9.9.1.3 of the IEEE 802.11e amendment and, $aRxTxTurnaroundTime$ is the maximum time (in μs) that the PHY requires to change from receiving to transmitting the start of the first symbol.

6.3 Timing Analysis

In this section, a worst-case timing analysis of the VTP-CSMA mechanism is presented, which demonstrates that the token rotation time is upper-bounded, even when the communication medium is shared with timing unconstrained stations. This means that the non real-time messages are not able to disturb the timing operation of the ring.

Consider an IEEE 802.11e network interconnecting *np* VTP-CSMA stations with multiple IEEE 802.11e stations (ST-stations). Consider that the VTP-CSMA stations have fixed addresses ranging from 1 to *np*. Each VTP-CSMA station accesses the network according to the VTP-CSMA scheme, *i.e.*, first station 1, then station 2, 3,... until station *np*, and then again station 1, 2,... *np*. The default (ST) stations implement the traditional backoff procedure according to the default timing values defined in [8].

Basically, two-collision scenarios are analyzed. Firstly, it is analyzed the maximum delay to transfer a real-time message, when the VTP-CSMA station is holding the token (Figure 6.6). According to the VTP-CSMA scheme, whenever a VTP-CSMA station holding the token has a data message ready to be transferred (D), it will wait an IFS (Interframe Space) before starting to transmit it (*1st* attempt). A station is able to detect a collision only after finishing its transmission plus an *aSIFSTime*, *i.e.* if the ACK frame is not received. Besides, when a transmission starts, all the stations set their NAV (Network Allocation Vector) with the information received in the Duration/ID field, that goes up to the end of the expected ACK frame [8]. Afterwards, if the transmission is not correctly acknowledged, the station will wait again during another IFS (IFS: $aSIFSTime + 2 \times aSlotTime$) interval and, according to the VTP-CSMA architecture, it will immediately start to transmit its message (*2nd* attempt). If a second collision occurs, the station A will wait again for the IFS before starting to transmit. The maximum time that a VTP-CSMA station holding the token will wait before starting to transfer a message for the last attempt or eventually discard it, is given by:

$$T_{col} = (RN - 1) \times (IFS + t_{message}) \tag{6.1}$$

where RN is the maximum retransmission number and $t_{message}$ is the duration to transfer a data message (including the ACK frame) from the VTP-CSMA station according to the physical (PHY) characteristics of the chan-

nel. For instance, considering 100 bytes for data payload in IEEE 802.11a PHY mode (data rate of 36 *Mbps*), each attempt takes $0.128ms$.

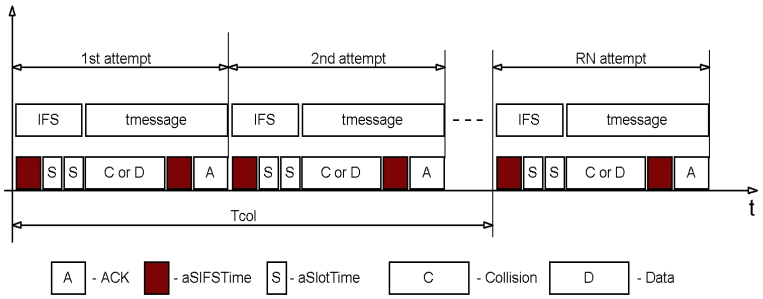

Figure 6.6: Collision scenario.

It is clear that the VTP-CSMA architecture either solves the collisions in a bounded time interval (within RN retransmission attempts), or it eventually discards the message. Therefore, a relevant focus of research is to evaluate the probability of a message frame being discarded by the IEEE 802.11 stations, whenever the number of collision resolution rounds exceeds the defined number of retransmission attempts. This topic has been addressed in [121], where it has been shown that for a non saturated network, the packet loss rate is smaller than 10^{-7}, when considering a collision probability $p = 0.1$. This satisfies the packet loss requirements of traditional soft real-time applications, such as VoIP (voice over IP) or NCS (networked control systems) applications.

Therefore, in the following analysis it is only considered the case where no message is discarded by a VTP-CSMA station, due to an excessive number of retransmission attempts. That is, when the transmitting station acquires the transmission medium, the virtual token will be passed to the following station at the end of the acquired TXOP (in the worst-case).

Then, the worst-case token holding time ($T_{TH_{RT}}$) occurs when the RT station has enough RT messages to fill up to the maximum TXOP interval defined for the subset of RT stations ($t_{TXOP_{RT}}$). The value of $T_{TH_{RT}}$ is given by:

$$T_{TH_{RT}} = T_{col} + t_{TXOP_{RT}} \tag{6.2}$$

On the other hand, when the VTP-CSMA station holding the token does not have any real-time message ready to be transferred, one of the ST

stations in the wireless domain can successfully transfer its own messages. This means that all the VTP-CSMA stations will wait during a time interval t_2, during which any ST station may start to transfer a message. In such a case, the ACo counters will be incremented only at the end of the acquired TXOP. If the medium remains idle during the t_2 time interval, the ACo counters will be immediately incremented (*i.e.*, there will be a Virtual Token Passing), allowing the next RT station in the ring to transfer its own messages.

The maximum time interval that a VTP-CSMA station is allowed to hold the token, when it does not have any real-time message ready to be transferred ($T_{TH_{NRT}}$), occurs when at the specific instant ($aSIFSTime + 2 \times aSlotTime$) a ST station takes the decision of transmit a frame and it acquires the transmission medium and uses all the allowed TXOP time. The value of $T_{TH_{NRT}}$ is given by:

$$T_{TH_{NRT}} = IFS + t_{TXOP} \tag{6.3}$$

Therefore, the worst-case token holding time T_{TH} is given by:

$$T_{TH} = max(T_{TH_{RT}}, T_{TH_{NRT}}) \tag{6.4}$$

It can be easily proved that $T_{TH_{RT}} > T_{TH_{NRT}}$, when both RT and ST stations have the same t_{TXOP}. However, most likely the value of $t_{TXOP_{RT}}$ for the RT stations will be set to a much smaller value than the default TXOP ($1.504ms$, for voice category) [8]. For example, considering that $t_{TXOP_{RT}} = 0.2ms$ and the maximum number of transmission attempts $RN = 4$, for the above described case (100 bytes for data payload and data rate of 36 Mbps), $T_{TH_{RT}} = 0.584ms$, while $T_{TH_{NRT}} = 1.547ms$. Therefore, it is expected that for most of the VTP-CSMA applications, the value of $T_{TH_{NRT}}$ will prevail.

Considering the token rotation time as the time interval between two consecutive token arrivals to a particular station, the worst-case token rotation time (TRT) is given by:

$$TRT = np \times T_{TH} \tag{6.5}$$

The value TRT imposes a lower bound for the periodicity of the real-time message streams supported by the VTP-CSMA architecture. The study of

the problem of guaranteeing synchronous message deadlines in the VTP-CSMA architecture can be easily adapted from [122], where a similar study was done for FDDI networks.

6.4 Performance Analysis

The target of the presented simulations is to evaluate how the VTP-CSMA architecture is able to support real-time (periodic) traffic in an open communication environment. The VTP-CSMA simulation model is implemented using a *Stochastic Petri Net* (SPN) model, previously developed to assess the timing behavior of the EDCA function [23]. This model was slightly modified in order to implement the VTP-CSMA architecture. The description of the EDCA model implementation is presented in Appendix A.

For all the simulations, it has been used a variant of the Gilbert-Elliot error model, where the channel is always in one of two states: Good or Bad. This model assume that bit errors are independent, with a fixed error rate in each state and, the state sojourn time is log-normal distributed.

According to Willig [117] this model is realistic for wireless transmission in an industrial environment. Then, for the parametrization of the used error model, it has been used the same values as defined in [117], *i.e.*, for all channels the mean duration of good state is $65ms$, the mean duration of bad state is $10ms$ and, the coefficient of variation (CoV) for the bad state holding times has been set to 10 and for the good state to 20. It is assumed that errors occur only in the bad state. Two sets of simulations are performed, differing in their respective mean bit error rate (BER). The first (*error-prone scenario*) set defines a mean BER of 10^{-4}, while the second (*error-free scenario*) set defines that no bit errors occur. Thus, during the bad channel states for the first set, the BER is about 0.00075 and, for the good state no bit errors will occur.

These mean burst lengths lead to a rather bad channel, where the steady-state probability for finding the channel in bad state is approximately 13.3%.

Furthermore, for the proposed simulations is assumed that there is no node mobility, nor hidden stations.

6.4.1 Simulation Scenario 1

The first simulation scenario is intended to assess the impact of the timing unconstrained traffic (generated by default stations) upon the real-time traffic. It considers an open communication environment (Figure 6.7), where multiple standard (ST) EDCA stations share the same communication medium with a subset of real-time (RT) stations implementing either the VTP-CSMA architecture (RT VTP-CSMA) or the EDCA mechanism (RT EDCA). Specifically, it is compared the VTP-CSMA architecture to the IEEE 802.11e EDCA mechanism using the highest access category (voice). The RT stations transfer just RT messages (small sized packets at periodic rate), whereas the ST stations transfer three types of traffic: voice (VO), video (VI) and background (BK).

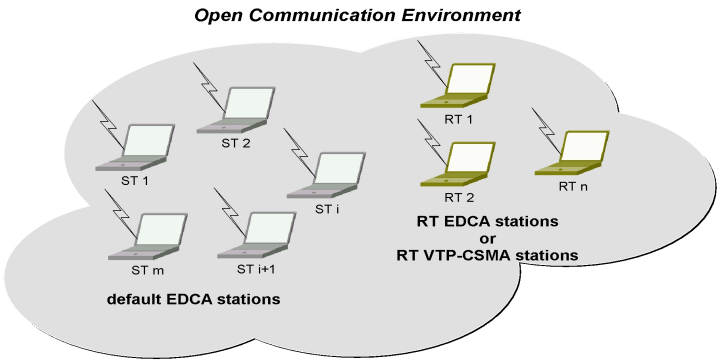

Figure 6.7: Simulation scenario.

Basically two simulation cases are analyzed. Firstly, a *small population* case considers 10 ST stations operating in the same frequency band together with 10 RT stations (either RT EDCA or RT VTP-CSMA). Secondly, the number of ST stations is extended to 40 (*large population*). Each station operates at OFDM PHY mode and the PHY data rate is set to 36 Mbps. The physical parameters used in the simulations are based on the IEEE 802.11a PHY mode [7] (Table 5.1).

The RT traffic is characterized by periodic sources with a small amount of jitter, modeled by a normal distribution with $\sigma/\mu \leq 1\%$ (σ is the standard deviation and μ is the average expected value). The ST traffic is modeled by Poisson traffic sources. The maximum number of transmission attempts is set to 4. The MAC queue size is set to 50 positions. All other relevant simulation parameters are shown in Table 6.1.

Table 6.1: Simulation data.

Parameters	**RT Station**		**Standard Stations**		
	EDCA	VTP-CSMA	VO	VI	BK
CW_{min}	7	0	7	15	31
CW_{max}	15	0	15	31	1023
AIFSN	2	2	2	3	7
TXOP (ms)	1.504	1.504	1.504	3.008	0
Packet Size - bytes	45	45	160	1280	1600
Message stream periods (ms)	2	2	Variable	Variable	Variable

The generated data frames have a constant size. Each RT station is transferring real-time data (with 45 bytes for data payload) with message stream periods of $2ms$, which is equivalent to generate 500 packets/s. Therefore, each RT station imposes a constant network load of 180 *kbits/s*, that represents about 0.5% of the total network load (without considering the MAC and PHY headers). For the set of ST stations, the offered load ranges from 5% to 95% of the total network load. Each ST station generates λ voice, video and background packets/s at the same rate, in order to impose the requested overall network load. The arrival rate (λ) can be obtained by:

$$\lambda = \frac{G_{ST}}{(PK_{VO} + PK_{VI} + PK_{BK})} = (packets/s) \tag{6.6}$$

where G_{ST} ranges according to the requested percentage (5% to 95%) of the PHY data rate (36 Mbps) and, PK_{VO}, PK_{VI} and PK_{BK} represent the packet size (bits) transmitted in each access category by the ST stations. The obtained value for the arrival rate λ is equally divided among all the ST stations.

6.4.2 Simulation Scenario 2

The second simulation scenario aims to verify how the VTP-CSMA architecture is able to cope with the tight requirements of real-time industrial communications [123], both in the presence of external timing unconstrained traffic and in the presence of error-prone channels. Specifically, it is of strong interest to assess the impact of external timing unconstrained traffic upon the *token rotation time*, since it bounds the shortest update time of periodic variables. Additionally, considering that the *transmission delay* is a good

estimation of the alarm latency[6], the assessment of the impact of external timing unconstrained traffic upon this parameter is also of paramount importance, as it must be kept smaller than the deadlines of the real-time message streams. Additionally, the average *queue size* is also assessed.

This simulation scenario considers multiple ST stations sharing the communication medium with a subset of RT stations implementing the VTP-CSMA mechanism. As in the previous scenario, the RT stations transfer just RT traffic. However, it is assessed the impact of the external network load upon the behavior of the VTP-CSMA architecture when a set of RT stations is transferring real-time data (45 bytes for data payload) with message stream periods of 10ms or 20ms (in the previous assessment only message stream periods of $2ms$ were considered). Basically two simulation cases are analyzed. The first scenario (also called *small population case*) considers 10 ST stations operating in the same frequency band together with 10 to 50 RT VTP-CSMA stations. The second scenario (*large population case*) extends the number of ST stations to 40.

All the other simulation parameters and details omitted in this subsection are equivalent to those used for the Simulation Scenario 1.

6.5 Simulation Results

All the simulation results have been obtained with 95% confidence interval with a half-width interval of 5%. The performance metrics analyzed in this chapter include: throughput, packet loss, average delay and average queue size. The *throughput* is the ratio between the total number of successfully transferred packets and the total number of generated packets for each traffic stream. Therefore, it represents the relative throughput. The *packet loss* metric is computed as *(1-throughput)*100* and represents the percentage of packets that are lost for each traffic stream. The *average delay* is the average delay required to transfer a packet, measured from the start of its generation at the application layer to the end of the packet transfer. The *average queue size* represents the average output buffer occupancy.

[6] The alarm latency is often considered a relevant performance index of factory communication systems [123].

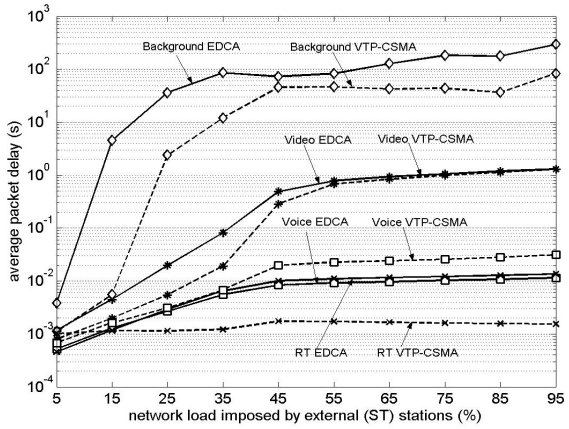

Figure 6.8: Average Delay - Small population (10RT - 10ST; $MSP = 2ms$).

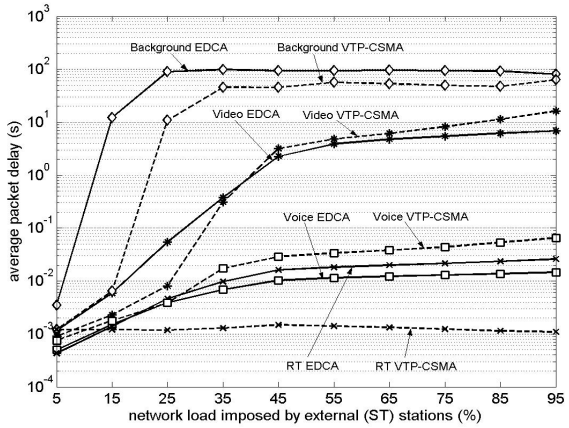

Figure 6.9: Average Delay - Large population (10RT - 40ST; $MSP = 2ms$).

6.5.1 Simulation Results: Scenario 1

The first simulation scenario is intended to assess the behavior of the VTP-CSMA *vs.* EDCA mechanisms, when supporting real-time traffic in a communication medium shared with timing unconstrained traffic. The average packet delays for the small and large population scenarios are plotted in Figures 6.8 and 6.9, considering an error-free channel. In both cases, the presented results are intended to compare the average delay for transferring a real-time (RT) packet using either the "VTP-CSMA mode" or the "standard EDCA mode".

Figures 6.8 and 6.9 show that, for an external disturbance above 15%,

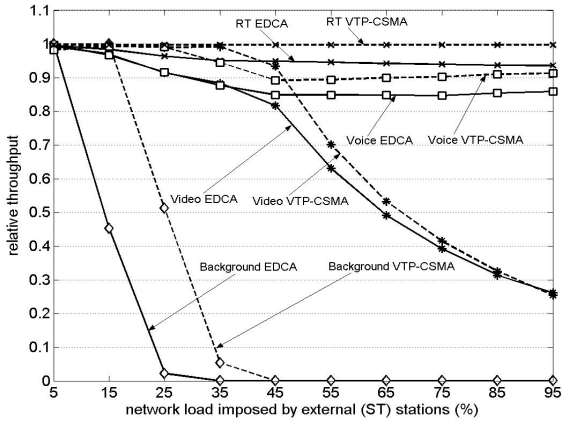

Figure 6.10: Throughput - Small population (10RT - 10ST; $MSP = 2ms$).

the RT VTP-CSMA stations have an average packet delay much smaller than the RT EDCA stations. For instance, in the large population Scenario (Figure 6.9), for a 55% external network load, the VTP-CSMA stations take in average $1.412ms$ to transfer a RT packet, while the EDCA stations take in average $18.161ms$. More importantly, these values must be checked against the message stream periods (MSP) of $2ms$. This means that the EDCA mechanism is clearly unable to provide a RT communication service to such message streams, whereas the VTP-CSMA approach is able to provide consistent average delays below the MSP value of $2ms$. This is a relevant result when considering real-time communications, as it clearly highlights the negligible impact of the timing unconstrained traffic upon the timing behavior of the real-time traffic. Figures 6.8 and 6.9 also illustrate that the VTP-CSMA stations have a small influence upon the behavior of the other access categories, except in what concerns a small (and expected) degradation of the voice (VTP-CSMA) traffic behavior.

The average throughput was also evaluated for, respectively, the small and the large population scenarios (Figures 6.10 and 6.11). The percentage of packet losses can be inferred from these results, as the percentage of packet losses is given by *(1-throughput)*100*. It is clear that RT VTP-CSMA stations are able to transfer significantly more RT packets than RT EDCA stations. The main reason is that, using the VTP-CSMA scheme, the RT packets generated by different RT stations are globally serialized, avoiding collisions among RT packets.

When considering the average packet delay, it can be also observed that

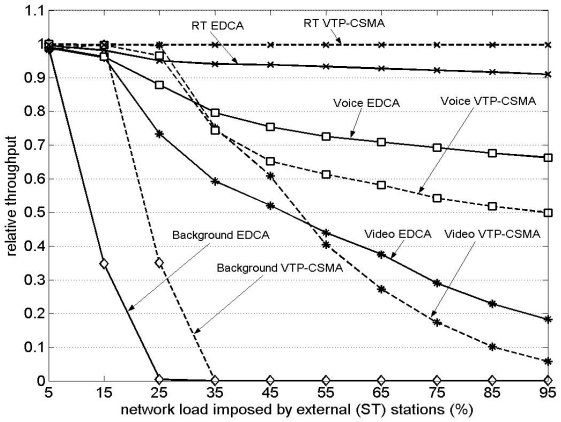

Figure 6.11: Throughput - Large population (10RT - 40ST; $MSP = 2ms$).

the increase of traffic load imposes a significant increase of the packet losses for lower priority traffics. For instance, in the large population case (Figure 6.11), for a network load higher than 25%-35%, a large number of background packets are discarded in both modes. From the simulation perspective, it is well known that smaller sample sizes lead to unreliable statistics values. Then, the average delay for background traffic is not relevant for values above this network load.

Another important aspect in RT communication is related to the average queue size (Figures 6.12 and 6.13). It can be seen that, for the transfer of RT packets, the VTP-CSMA stations have an average queue size much smaller than the RT EDCA stations. It is also clear that, whatever the network load, the average queue size of the VTP-CSMA stations is nearly constant and smaller than 1. This means that, in the average case, the pending RT messages are always sustainable transferred before the generation of new RT messages, meaning that the deadlines are respected. This is a very relevant result for real-time communications, as it forecast an adequate timing behavior for the VTP-CSMA architecture.

The results presented in Figures 6.12 and 6.13 are similar to the results presented in Figure 5.5, thus allowing for direct comparison. Both scenarios consider 10 RT stations together with 10/40 ST stations, where the RT stations are supporting RT message streams with $MSP = 2ms$. The simulation data are similar for both cases (Table 5.2 *vs.* Table 6.1), thus the results can be directly compared.

As it was also expected, the average queue size for voice packets is smaller

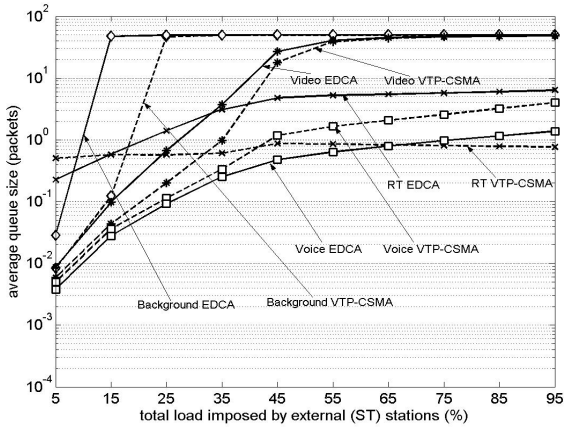

Figure 6.12: Queue Size - Small population (10RT - 10ST; $MSP = 2ms$).

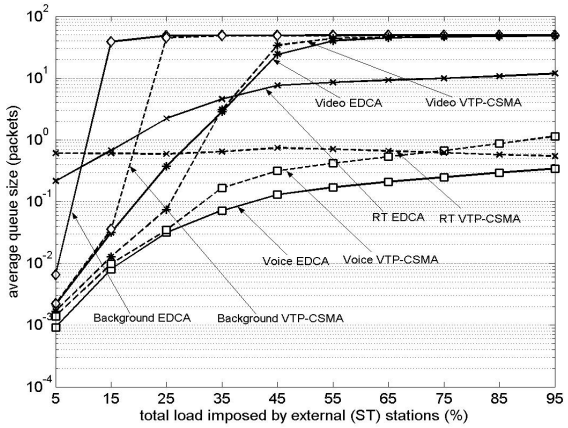

Figure 6.13: Queue Size - Large population (10RT - 40ST; $MSP = 2ms$).

than the average queue size for RT traffic, as the RT traffic load is much larger than the load imposed by the VO traffic. Whatever the network load, each RT station imposes a packet rate of 500 packets/s, while each ST station imposes just (in the worst case: small population scenario at 100%) a voice packet rate of 140 packet/s.

Finally, as the error-prone behavior of wireless channels is a relevant drawback when supporting real-time communications, it is important to assess the impact of the error-prone channels upon the behavior of the VTP-CSMA architecture. Therefore, a set of simulation experiments have been carried out, considering a highly pessimistic scenario for the channel error parameters. For the simulated scenarios, it has been used a $BER =$

7.5×10^{-4} and a mean duration for the Bad state of $10ms$. Therefore, as each RT station generates 1 packet every $2ms$, during the burst error periods ($10ms$) the successful transmission probability to transfer a RT packet is just of 60%[7]. For the sake of simplicity, only the values for RT stations (EDCA or VTP-CSMA) are plotted for Scenario 1.

Figures 6.14 and 6.15 illustrate the average packet delay, in the small and large population scenarios, considering both error-free and error-prone channels. In spite of being nearly constant ($3ms$ - $5ms$), the average delay is significantly larger than for the case of an error-free scenario ($1ms$ - $1.6ms$). This behavior can be rooted to two different causes. On the one hand, whenever the distributed variable ACo has no longer a consistent value, sooner or later the reset mechanism will be called up (Initialization procedure) to re-synchronize the virtual ring. This mechanism is highly effective, as the average packet delay remains nearly constant despite the external network load. On the other hand, the considered packet error rate is highly pessimistic (40% of packet error probability during the error bursts). Therefore, the re-synchronization rate of the virtual ring is much higher than the expected for traditional industrial environments.

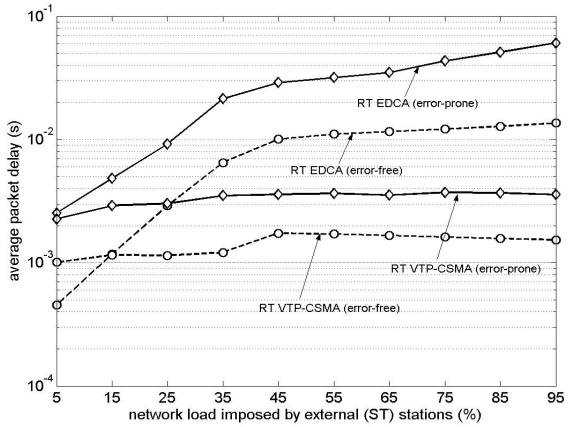

Figure 6.14: Average Packet Delay - Small population (10RT - 10ST; $MSP = 2ms$).

Another important aspect in RT communication is related to the average queue size. From Figures 6.16 and 6.17, for an error-free channel, the average queue size of the RT VTP-CSMA stations is nearly constant and smaller

[7] Such probability is equal to $(1 - BER)^n$, where n is the total length (bits) of a RT packet.

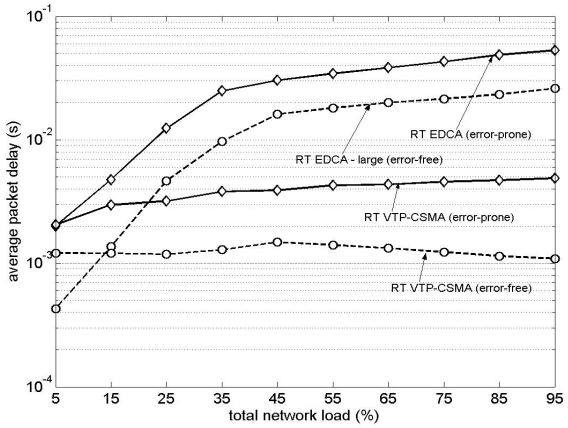

Figure 6.15: Average Packet Delay - Large population (10RT - 40ST; $MSP = 2ms$).

than 1. This result is consistent with the packet generation periodicity of $2ms$ and the average packet delay of ($1ms$ - $1.6ms$). For the error-prone channel, as the average packet delay is ($3ms$ - $5ms$), the average queue size will be larger than 1 (thus, there will be frequent deadlines misses). This means that the VTP-CSMA approach has reached its usability bounds.

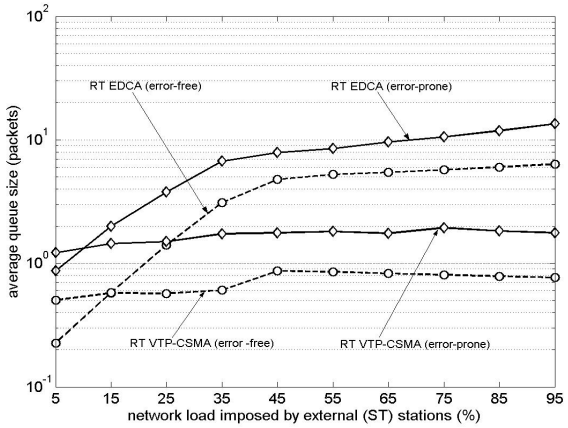

Figure 6.16: Queue Size - Small population (10RT - 10ST; $MSP = 2ms$).

6.5.2 Simulation Results: Scenario 2

The performance analysis results presented in this subsection aims to verify how the VTP-CSMA architecture is able to cope with the tight requirements

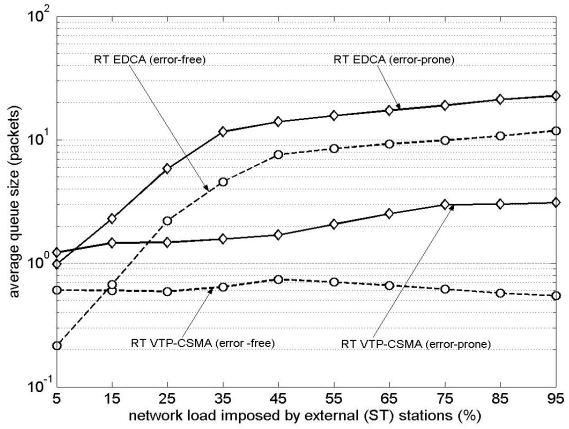

Figure 6.17: Queue Size - Large population (10RT - 40ST; $MSP = 2ms$).

of real-time industrial communications, both in the presence of external timing unconstrained traffic and in the presence of error-prone channels.

Firstly, it is evaluated the token rotation time, as the ring stability cannot be affected by the external traffic sources with unconstrained timing behavior. We have assessed the behavior of the token rotation time for different real-time configurations (10, 20 and 50 RT stations), when supporting real-time message streams with periods of $10ms$ or $20ms$ (in the previous scenario, only 10RT stations with message stream periods of $2ms$ were considered). Figures 6.18 and 6.19 illustrate the impact of external timing unconstrained traffic upon the token rotation time for message stream periods of $10ms$ an $20ms$, respectively. It is plotted the token rotation time in both error-free (solid lines) and error-prone (dashed lines).

As expected, the token rotation time have a reduced value while the network load is kept low. The main reason is that the medium remains idle during long time intervals, and therefore each RT station will capture the token almost always after t_2 ($aSIFSTime + 3 \times aSlotTime$). This means that the probability of having a ST station contending for the medium access is small. When the network load increases, whenever a RT station does not have any RT message to transfer, it is expected that *one* ST station will be able to capture the transmission medium during *one* TXOP time interval.

In Figures 6.18 and 6.19, it is also plotted the upper-bound for the TRT obtained from Equation 6.5. It must be considered that this upper-bound addresses a rarely occurring case, as it is based on the assumption that during a token rotation cycle, none of the RT station had RT-messages to

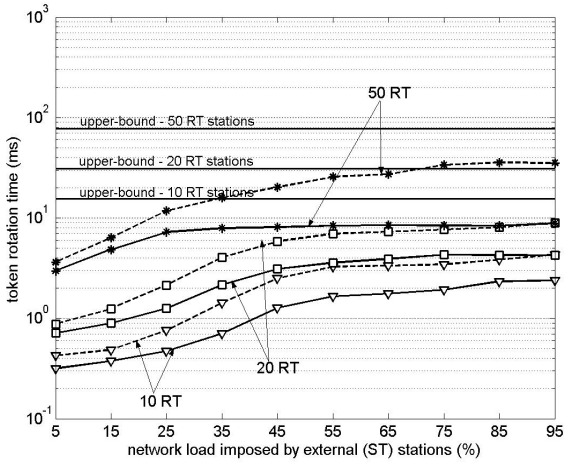

Figure 6.18: Token rotation time (10/20/50RT - 40ST; $MSP = 10ms$).

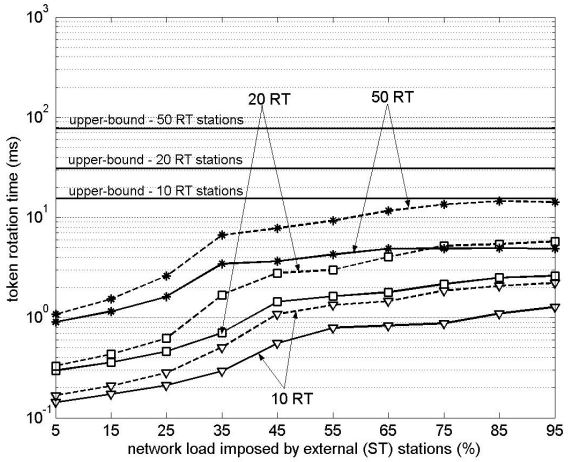

Figure 6.19: Token rotation time (10/20/50RT - 40ST; $MSP = 20ms$).

transfer, and that every ST-station transferred messages up to the maximum allowed TXOP (1.504ms for voice category).

From Figures 6.18 and 6.19, it can be concluded that the VTP-CSMA architecture has a stable ring operation for message stream periods of $10ms$ and $20ms$, *i.e.*, it guarantees a stable token rotation time, whatever the timing unconstrained traffic load (in both error-free and error-prone channels). Therefore, the VTP-CSMA architecture is able to provide a RT communication service to the supported real-time applications.

A second simulation analysis concerns the assessment of the average

queue size in a RT station, when the traffic in the wireless domain is disturbed by the presence of timing unconstrained traffic from ST stations and from physical interferences (error-prone channel). Figure 6.20 shows the average queue size for MSP of $10ms$ in both error-free (solid lines) and error-prone (dashed lines) channels, where it is plotted the cases of 10, 20 or 40 RT stations operating in the small population scenario (10 ST stations). It is clear in Figure 6.20 that the average number of packets waiting to be transmitted is always kept under 1 packet. This indicates that the VTP-CSMA mechanism can be suitable to support real-time traffic with MSP of $10ms$, as the pending RT messages are always transferred (in average) before the generation of new RT messages. Similar results were obtained for MSP of $20ms$ in both small and large population scenarios.

When comparing these results (for 20 RT stations), with the results obtained from the EDCA mechanism (illustrated in Figure 5.6) the following conclusion arises: while the VTP-CSMA approach guarantees almost constant values for the average queue size (small range between 0.3 - 0.4), the EDCA function (for 20RT stations - 10 ST stations) shows a wider range for the average queue size values (0.2 up to 0.88). This behavior illustrates the smaller sensitivity to the external load variations provided by the VTP-CSMA approach when compared to the EDCA mechanism.

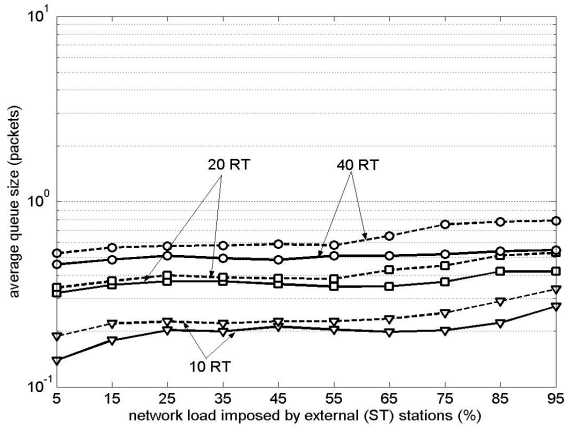

Figure 6.20: Average queue size (10/20/40RT - 10ST; $MSP = 10ms$).

Therefore, the VTP-CSMA architecture is able to provide an adequate RT communication service to soft real-time applications, such as VoIP or NCS applications. Voice (and video) conversation need to be conducted with the minimum of delay in transmission. Many VoIP systems use message

stream periods (MSP) of $20ms$ and, an average packet delay below $150ms$ is acceptable for most user applications [13]. Besides, it is recommended a packet loss rate $< 2\%$ and average jitter $< 50ms$, what is easily reached by the VTP-CSMA architecture.

6.6 Virtual Ring Management

The initial version of the VTP-CSMA architecture presented in Section 6.2 was able to handle only a fixed number of RT stations (*np* stations). In this section, an enhanced ring management procedure is proposed, allowing the VTP-CSMA architecture to be an *open* group. Thus, a station can dynamically join or leave the Virtual Token Ring (group G), enabling the support of dynamic communication scenarios, such as those that are usually found in VoIP applications. The ring management includes procedures to (i) add RT-stations to the ring, (ii) remove RT-stations from the ring. These procedures must ensure the two following properties.

Agreement: All VTP-CSMA stations must agree on the values of ACo and *np*. That is, at whatever instant of time, all stations know the address of the token holder and the total number of stations belonging to the group G.

Uniqueness: Each station must be assigned an unique NA, which ranges between 1 and *np*.

Unless these two properties are satisfied, mutual access to the medium by the RT-stations cannot be ensured. Besides these properties, some further assumptions are made regarding the capabilities of the communication system.

The first assumption is that stations belonging to group G are able to exchange messages. Each message (*msg*) contains the field *sender* and *type*; *sender* identifies the source, whereas *type* is concerned with the function of the message itself. Depending on the value assigned to the *type* parameter, messages may be used either to transfer real-time data or to manage the group membership. The field *type* may assume the following values {REMOVE, JOIN, ADD, UPDATE, HB, RT}:

- *msg.JOIN*, *msg.ADD* and *msg.UPDATE* messages: are used to add a station to the group;

- *msg.REMOVE* message: is used for removing a station;
- *msg.HB* message: is used to send a "heartbeat message" and;
- *msg.RT* message: are default RT data frames.

The second assumption specifies that the logical ring has already been initialized with some VTP-CSMA stations. Finally, it is desirable that all VTP-CSMA stations remain in the range of the Access Point (AP); conversely, for *ad hoc* networks, it is desirable that stations can always hear each other. However, both the hidden and exposed scenarios are considered in the protocol specification. Communication errors can occur due to collisions and/or interferences.

6.6.1 Adding a Station to the Virtual Token Ring

A station willing to join the VTR, will broadcast a *msg.JOIN* message using the default EDCA mechanism. As a consequence, the message will be received by all the network stations, but queued only by those belonging to group G (all the other stations will discard it). However, only one response has to be issued to the requesting station. It will be provided by station NA_1, via the *msg.ADD* message[8]. Therefore, when $ACo{=}NA_1$, the token holder station will allow the requesting station to join the group through an acknowledged point-to-point control message of type *msg.ADD*, with the following parameters:

ACocurr: $\ll integer \gg$ - Current value of the Access Counter.

NAA: $\ll integer \gg$ - NA assigned to the requesting station: $NAA \leftarrow NP{+}1$.

MAC: $\ll HEX \gg$ - MAC address of the requesting station.

Then, the requesting station (*i.e.* the station that issued the *msg.JOIN*) sets $ACo \leftarrow ACocurr$, $NA \leftarrow NAA$, $NP \leftarrow NAA$ and enters the group G. The MAC address of the requesting stations will be used if more than one join request arrives. In this case, station NA_1 has to answer separately to each

[8] It is worth noting that, in an infrastructure network, station NA_1 will be, most likely, the Access Point.

station that issued the request. The value of the MAC address may be obtained from the *msg.JOIN*. After each *Adding* procedure, the membership can be represented as $L = \{NA_1, NA_2, ..., NA_{np}, NA_{np+1}\}$.

After processing all the received join requests, in order to inform all the VTP-CSMA stations of the new entry, the token holding station (NA_1) broadcasts an unacknowledged message of type *msg.UPDATE*, with the following parameters:

ACocurr: $\ll$ *integer* $\gg$ - Current value of the Access Counter.

NPcurr: $\ll$ *integer* $\gg$ - Current value of the number of stations.

Consequently, all stations belonging to the group G will update the value of their VTR parameters ($ACo \leftarrow 1$, if different from the previous one; $NP \leftarrow NPcurr$). Setting the value of all the ACo counters to 1 ensures the re-synchronization of potentially inconsistent values of the ACo distributed variable (if any), when compared to station NA_1.

6.6.2 Removing a Station from the Virtual Token Ring

There are two possibilities for removing a station from the virtual token ring (group G). In the first, the station decides autonomously to leave the group G. In this case, it will communicate its decision via a remove message (*msg.REMOVE*), whenever it receives the virtual token. In the second, the station is compelled to leave the VTR, either due to a "crash failure" or because it becomes unable to transfer its own VTP messages. As an immediate effect, there will be no more RT messages in the slot assigned to that station. In such a case, the unused NA address will be later recovered by an *address reclaim* procedure (based on a "heartbeat" approach).

In order to autonomously leave the group G, the station holding the virtual token broadcasts an unacknowledged *msg.REMOVE message*, which has the following parameters:

NAR: $\ll$ *integer* $\gg$ - NA to be removed from the VTR.

As a consequence, all the VTP-CSMA stations that receive a *msg.REMOVE* will update their variables in the following way:

1. stations with $\mathrm{NA}_i < NAR$:
   (a) update their NP ($NP \leftarrow NP\text{-}1$);
2. stations with $\mathrm{NA}_i > NAR$:
   (a) update their NP ($NP \leftarrow NP\text{-}1$);
   (b) update their NA_i ($NA_i \leftarrow NA_i\text{-}1$);
3. stations with $NA = NAR$:
   (a) The stations are considered to be out of the ring;

The latter may occur only in the case of an inconsistent ACo distributed variable. In such a case, the station must consider itself out of the ring. If it wants to re-enter the ring, in must later "rejoin" it. As a further check, stations belonging to group G may verify the correctness of their image of the distributed variable ACo (Access Counters). Indeed, it should be $NAR{=}ACo$ for each station. After the removal of a station, the membership of group G can be represented as $L = \{NA_1, NA_2, ..., NA_{np-1}\}$.

A VTP-CSMA station, may also leave the VTR without being able to broadcast the *msg.REMOVE* message (station crash). Such an event has first to be detected and then the network address of the station (NA) must be recovered (via the address reclaiming procedure), in order to ensure the consistency of the ring. The detection procedure works as follows. In the VTR, each station knows the address of its predecessor ($PS{=}\{(i\text{-}1) \bmod NP\}$). Whenever a station detects that its predecessor has not sent any message during a T_{live} time interval, it concludes that the station has unexpectedly exited the virtual ring. Thus, a station detecting such an exit, invokes the *address reclaiming* procedure. In particular, it broadcasts a *msg.REMOVE* message to remove its predecessor from group G.

It is worth mentioning that, in order to avoid wrong removals, every station must transfer at least one message every T_{Live}. The technique adopted to avoid wrong removals uses a heartbeat approach, where the $T_{HB_{own}} = \frac{1}{2}T_{Live}$ (lines 23-28, Figure 6.22). In practice, a station that has not sent any message during the last $\frac{1}{2}T_{Live}$ interval, the next time it gets the virtual token, it will transfer either a *heartbeat* message (*msg.HB*) or a real-time message (*msg.RT)*.

```
1  if ACo = NA_{i-1} then
2      T_{HB_pred} ← 0;
3  endif

4  if AC = NA_{i-1} then
5      start T_{HB_pred};
6  endif
```

Figure 6.21: Additional Listening items.

6.6.3 Implementation Details

In order to implement the Adding and Removing procedures, it is necessary to add some functionalities to the original procedures. This section describes the main modifications that must be done to allow real-time stations to dynamically join or leave the VTP-CSMA architecture. Firstly, to implement the address reclaiming procedure, it has been necessary to include a $T_{HB_{pred}}$ timer (heartbeat timer) in the *Listening* procedure, which will be reset each time the predecessor station (PS) sends a message. Therefore, it is necessary to include lines 1-3 from Figure 6.21 after line 7 of the *Listening* procedure (Figure 6.4). Furthermore, whenever $ACo = NA_{i-1}$ and the ACo value is incremented by *continuing_idle* state (t_2), it indicates the PS station does not send any message and $T_{HB_{pred}}$ timer must be started. Then, it is necessary to include lines 4-6 from Figure 6.21 after line 16 of the *Listening* procedure (Figure 6.4). Besides, the variable t_3 is no longer necessary and, consequently lines 18-19 from *Listening* procedure must be excluded. This is one of the advantages of the ring management procedure. Whenever a VTP-CSMA station becomes out of order (*e.g.* when it did not receive an unacknowledge control message), the station will consider it self out of the ring, and will later re-join it (Figure 6.22, lines 16-17). This means that there is no longer the need to re-initialize the VTP-CSMA architecture from scratch (as it was the case in the static version of the VTP-CSMA approach).

The main modification to implement the ring management proposal must be done in the *Transmission* procedure. For the sake of cleanness, this procedure is rewritten incorporating such additional modifications (Figure 6.22). One of the modifications considers the case when the token holding station detected that its PS did not send any message during a T_{Live} interval. In such a case, it sends a *msg.REMOVE* message removing its predecessor

```
1  Transmission:
2  if (ACo=NA_i=1) then
3      while NA_i has msg.JOIN buffered do
4          send msg.ADD;
5      end while
6      send msg.UPDATE
7  endif
8  if T_HB_pred ≥ T_Live then
9      send msg.REMOVE;
10 endif
11 if NA_i has msg.RT to be transmitted then
12     T_HB_own ← 0; send msg.RT;
13     if successful transmission then
14         t_1← 1; go to Listening;
15     else
16         if too many attempts then
17             NA_i do "rejoin";
18         else
19             go to Transmission;
20         endif
21     endif
22 else
23     start T_HB_own;
24     if T_HB_own ≥ 1/2 T_Live then
25         send msg.HB; t_1← 1; T_HB_own ← 0; go to Listening;
26     else
27         t_2← 2; go to Listening;
28     endif
29 endif
```

Figure 6.22: Transmission procedure (ring management).

(lines 8-10). On the other hand, if the token holding station NA_i inside of group G had no message to transmit during $\frac{1}{2}T_{Live}$, it will send a *msg.HB* message in order to avoid wrong removals (lines 24-25).

In case multiple stations issue a request to join the VTR, there is the need to distinguish between the subsequent *msg.ADD* messages that have to be delivered, possibly, to different stations, in response to the join requests. This requires to send, as a parameter, the MAC address of the requesting station (lines 2-7).

Finally, in the *Initialization* procedure it is necessary to initialize the variable $T_{HB_{own}}$ and $T_{HB_{pred}}$ and, the *Main* procedure must be executed whenever the VTP-CSMA architecture is running, instead of while $t_3 \leq RN$.

6.6.4 Validation of the VTP-CSMA Architecture

In order to validate the proposed VTP-CSMA architecture, one major issue needs to be investigated: the "structural" behavior of the VTP-CSMA architecture.

One of the weaknesses of the proposed ring management scheme is rooted to the use of unacknowledged broadcast messages. The loss of the *msg.REMOVE* and/or *msg.UPDATE* messages by some nodes may lead to the violation of any, or even both, of the two correctness properties (Agreement and Uniqueness). For example, if one of the VTP-CSMA stations do not receive a *msg.REMOVE* message, then it will not update its distributed variables (*NA*, *ACo*, *NP*). As a consequence, there will be an inconsistency in the distributed variables, and sooner or later there will be the collision of RT-messages which, in turn, forces both stations to consider themselves out of the ring. This means that the structural behavior of the VTP-CSMA architecture is ensured by means of a self-removal mechanism that removes the two colliding stations. Thus, it is expected that its impact upon the performance of the ring remains negligible (as just the inconsistent station plus *one* consistent station are removed from the ring). That is, conversely to other token passing schemes, the ring does not needs to be re-built from scratch.

The proposed ring management procedure has to be carefully assessed via an adequate performance analysis to validate the effectiveness of the self-removal reset mechanism when dealing with inconsistent distributed variables. Preliminary results of the performance analysis have already highlighted the effectiveness of such "self-removal" approach upon the original *ACo* reset mechanism.

6.7 Summary

The major motivation for this work was to "propose a solution enabling the support of real-time (RT) communication in wireless IEEE 802.11 environments, where timing unconstrained devices would be able to coexist with real-time devices". The Virtual Token Passing VTP-CSMA approach has been proposed to target this problem.

The initial version of the VTP-CSMA architecture has been proposed to support real-time communication in IEEE 802.11e wireless networks with a fixed number of real-time stations (static environment).

The timing analysis carried out in this chapter demonstrates that the token rotation time of the VTP-CSMA architecture is upper-bounded, even in the presence of external timing unconstrained traffic. This means that the VTP-CSMA architecture is able to provide a real-time communication service, even when the communication medium is shared with timing unconstrained traffic sources. The performance analysis that have been carried out highlights the adequate behavior of the VTP-CSMA mechanism in error-prone channels.

The obtained simulation results clearly demonstrate that the proposed mechanism guarantees the highest transmitting probability to VTP-CSMA stations in a wireless environment, where the communication medium is shared with timing unconstrained traffic sources. More importantly, whatever the network load, both the average packet delay and related average queue size are nearly constant for the RT traffic. This means that the timing unconstrained network traffic has a negligible impact upon the timing behavior of the RT traffic.

A Ring Management procedure for the VTP-CSMA architecture has also been proposed, that allows real-time stations to dynamically join or leave the VTP-CSMA architecture. This ring management procedure extends the originally proposed VTP-CSMA architecture, enabling its operation in dynamic scenarios, such as those found in state-of-the-art VoIP applications.

Chapter 7

Conclusions and Future Work

This chapter summarizes the major research results achieved throughout this thesis, highlighting how the research contributions fulfilled the original research objectives. Furthermore, some guidelines are given for future research directions that may emerge from this work.

7.1 Conclusions

The major motivation for the research work presented in this thesis was to "propose novel solutions that enable the support of real-time (RT) communication in CSMA-based networks". Within this context, several approaches to support RT communication in CSMA-based networks have been presented, where a special emphasis was given to solutions compatible with IEEE standard wired and wireless networks. The main focus of the research work that have been carried out during this thesis was the analysis and design of communication solutions intended to support real-time communication services in shared communication environments.

First of all, a survey of the state-of-the-art on real-time communication in CSMA-based networks has been made and an innovative classification framework has been proposed. A *first classification axis* related to how message collision are dealt with classifies the researched solutions into the following groups: avoiding collisions; solving collisions or; reducing collisions. A *second classification axis* highlights how the state-of-the-art proposals keep or alter the compatibility with IEEE 802.3/802.11 compliant

devices. Throughout this survey, it is clear that few techniques allow standard (non real-time) stations to coexist with enhanced (real-time) stations in the same communication environment (compatibility levels 2 and 3). The majority of nowadays solutions require the full environment control, *i.e.* all the network stations must be under the strict control of the real-time communication architecture (compatibility level 1).

Considering now the case of wireless networks, it is not realistic to assume that is possible to create a zone free of wireless stations, as the wireless channel is essentially a shared communication environment. Thus, approaches that guarantee the RT behavior through the tight control of every communicating device are no longer applicable, as those approaches will not be able to support RT communication services adequate for the next generation of communication environments. That is, communication environments characterized by an unpredictable traffic load and an unpredictable number of communicating devices. Besides, most of those approaches proposed for both Ethernet and WiFi networks cannot be implemented using COTS hardware, consequently, they can be technically very interesting, but not economically viable today. Thus, the following assumptions/requirements are made in this thesis:

- the target communication environment follows either the IEEE 802.3 or the IEEE 802.11 communication standards, with default stations (ST stations) and real-time stations (RT stations) sharing the same communication medium (wired or wireless);

- the network load imposed by the set of ST stations is out of the sphere-of-control of the RT communication architecture;

- no hardware/software changes should be necessary to ST stations, due to the large installed base of IEEE 802.3/802.11 devices.

None of the existing state-of-the-art protocols efficiently fulfills all these assumptions/requirements. We have pointed out that the most promising solutions to support RT communication are those that force the collision resolution in favor of the RT station. Therefore, we concluded that a new *paradigm* will emerge to deal with real-time communications in open communication environments, which will be based on a *forcing collision resolution* approach.

Within this context, a new RT-communication approach (VTP-CSMA approach) has been proposed, which is based on the use of traffic separation mechanisms. Such mechanisms are able to prioritize RT-traffic over multipurpose traffic, without directly controlling the latter. That is, instead of controlling *all* the traffic generated by *all* the stations, which is an approach ruled out by the identified set of assumptions/requirements, the proposed VTP-CSMA approach controls only the traffic generated by the RT stations. This approach enables the fulfillment of the RT communication requirements in open communication environments.

Specifically, we propose a new Traffic Separation mechanism (TSm) (chapter 3) that prioritizes real-time traffic over timing unconstrained traffic. This mechanism is used as an underlying traffic separation mechanism, that will enable the provision of RT communication services in CSMA-based architectures. The proposed TSm mechanism has been assessed both by simulation and probabilistic analysis, which demonstrate that it guarantees the highest transmitting probability for a TSm-enabled station in any shared communication environment. The resulting average access delay is significantly smaller, when compared with the access delay for the standard stations. More importantly, it is clear that, whatever the network load imposed by external (non real-time) stations, the average packet delay for the real-time traffic is nearly constant. Therefore, it has been shown that the proposed traffic separation mechanism enables the implementation of a simple "forcing collision resolution" scheme in favor of RT stations using COTS hardware (compatibility level 3).

After analyzing the suitability of the proposed TSm mechanism to support real-time communication, it has been presented and evaluated two innovative architectures: the VTPE-hBEB and the VTP-CSMA architectures. Both architectures are based on the control of the medium access right, by means of a virtual token passing procedure among real-time stations, complemented by the use of the TSm mechanism. One of the main advantages of these proposals is that, they allow the coexistence of IEEE 802.3 or IEEE 802.11 standard (non real-time) stations with multiple enhanced (real-time) stations, thus enabling the fulfillment of of the real-time requirements associated to the next generation communication environments.

The VTPE-hBEB architecture imposes a higher priority for the transfer of VTPE-hBEB related traffic, guaranteeing the required traffic separation. The analysis included in chapter 4 shows that, for a moderate number of nodes, the token rotation time is of the order of a few milliseconds. This

figure seems adequate for real-time applications in the automation domain. Delivery times below $10ms$ (class 2 communication requirements) can easily be reached by VTPE-hBEB mechanism and, with some effort it is possible to reach delivery times around $1ms$ (class 3).

The VTP-CSMA architecture (chapter 6), which is the main contribution of this thesis, proposes a similar approach for IEEE 802.11 networks. It considers an unified wireless system operating in one frequency band, with the communication bandwidth shared by real-time and non real-time communicating devices. The obtained simulation results clearly demonstrate that the proposed mechanism guarantees the highest transmitting probability to VTP-CSMA stations in wireless environments. More importantly, whatever the network load imposed by external ST stations, both the average packet delay and related average queue sizes are nearly constant for the RT traffic. This means that the unconstrained network traffic has a negligible impact upon the timing behavior of the RT traffic. Additionally, it has also been assessed the capability of the VTP-CSMA mechanism to handle a large amount of RT stations in error-free and error-prone scenarios. The obtained results clearly show that even for more stringent scenarios, the VTP-CSMA architecture is able to support a number of stations compatible with traditional real-time communication environments supporting either VoIP or NCS applications.

A Ring Management procedure for the VTP-CSMA architecture has also been proposed, allowing real-time stations to dynamically join or leave the VTP-CSMA architecture. This ring management procedure extends the originally proposed VTP-CSMA architecture, enabling its operation in dynamic scenarios, such as those found in state-of-the-art VoIP applications.

Finally, a Stochastic Petri Net (SPN) simulation model have been implemented, enabling the evaluation of IEEE 802.11e networks. The implemented model focus on the behavior of the Enhanced Distributed Channel Access (EDCA) mechanism. The implemented SPN model was built in order to understand the limitations of the EDCA mechanism, when it is used to support RT communication services (chapter 5). Thus, it was assessed the suitability of the IEEE 802.11e EDCA protocol to support industrially-relevant real-time communication scenarios in open communication environments. The main conclusion from the simulated scenarios was that when the voice priority is used to support real-time traffic in shared medium environments, the default parameter values of the EDCA mode are not able to guarantee the industrial communication (timing) requirements. Moreover,

it is clear that the major impact over the RT communication is due to collisions, instead of the error-prone characteristics of wireless medium. That is, there is a common misinterpretation that major impact on communication derives from interferences, *e.g.* Electromagnetic interferences.

7.2 Future Work

The future work that arise from this thesis is closely related to the VTP-CSMA architecture. Firstly, the proposed ring management procedure has to be carefully assessed via an adequate performance analysis to validate the proposal. The ring management behavior is closely related to the use of unacknowledged broadcast messages. The loss of these messages by some nodes may lead to the violation of any, or even both, of the two correctness properties (Agreement and Uniqueness). Therefore, it is necessary to evaluate the impact of the loss of broadcast messages upon the ring timing properties. It is expected that its impact remains negligible (as just the inconsistent station plus *one* consistent station are removed from the ring).

The VTP-CSMA architecture could be also analyzed from the fault-tolerance perspective. To overcome the drawback of broadcast messages, for example, it can be evaluated/proposed techniques providing reliable broadcast message deliveries. Also, as the station "NA_1" is a single point of failure, it would be interesting to analyze the time that the VTP-CSMA architecture takes to recovery from a failure. Summing up, a formal analysis with a fault-tolerance perspective seems adequate. This analysis could be used to identify the major failures of the protocol and, improvements should be proposed.

Another research perspective is that the VTP-CSMA architecture considers only one virtual ring. However, in real environments, it is possible for a station to listen to more than one ring. Therefore, an important research work would be to propose a multiple ring management, allowing wireless multi-hop RT communication.

Preliminary research conclusions report that the HCCA mechanism is not adequate to support real-time communication. However, the HCCA mechanism may have the potential to provide an interesting real-time communication service. Moreover, there are some recent proposed scheduling algorithms operating under HCCA intended to improve the original HCCA mechanism and to provide RT communication. Therefore, it would be

also interesting to compare the VTP-CSMA architecture with the standard HCCA mechanism and its improvements.

Another important research work is the study of the problem of guaranteeing synchronous message deadlines in the VTP-CSMA architecture. A similar study [122] was done for FDDI networks. Those studies could be easily adapted to the proposed VTP-CSMA architecture.

Bibliography

[1] P. Ramanathan, "Overload management in real-time control applications using (m, k)-firm guarantee," *IEEE Transactions on Parallel and Distributed Systems*, vol. 10, no. 6, pp. 549 – 559, 1999. [cited at p. 1]

[2] T. Sauter and F. Vasques, "Editorial: Special section on communication in automation," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 2, pp. 73 – 77, May 2006. [cited at p. 2, 76]

[3] B. Tavli and W. Heinzelman, *Mobile Ad Hoc Networks*. Dordrechtf: Springer, 2006. [cited at p. 2]

[4] "IEEE Standard for Information Technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements. part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," ANSI/IEEE Std 802.3, 1985. [cited at p. 2]

[5] J.-D. Decotignie, "Ethernet-based real-time and industrial communications," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1102 – 1117, 2005. [cited at p. 2, 3, 37, 40, 79]

[6] M. Felser, "Real-time Ethernet - industry prospective," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1118 – 1129, 2005. [cited at p. 3, 76]

[7] "IEEE Standard for Information Technology - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," ANSI/IEEE Std 802.11, 1999 Edition (R2003), 2003. [cited at p. 3, 9, 25, 48, 127, 182, 184, 185]

[8] "IEEE Standard for Information Technology - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements," IEEE Std 802.11e-2005, 2005. [cited at p. 3, 25, 37, 48, 123, 125, 177, 180, 182, 184, 185, 186, 187, 190]

[9] A. Willig, K. Matheus, and A. Wolisz, "Wireless technology in industrial networks," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1130 – 1151, 2005. [cited at p. 3, 40, 98, 100]

[10] H. Kopetz, "Time-triggered model of computation," *Real-Time Systems Symposium*, pp. 168 – 177, 1998. [cited at p. 3]

[11] D. Miorandi, E. Uhlemann, S. Vitturi, and A. Willig, "Guest editorial: Special section on wireless technologies in factory and industrial automation, part i," *IEEE Transactions on Industrial Informatics*, vol. 3, no. 2, pp. 95–98, 2007. [cited at p. 4, 38]

[12] M. Hamdaoui and P. Ramanathan, "A dynamic priority assignment technique for streams with (m,k)-firm deadlines," *IEEE Transactions on Computers*, vol. 44, no. 12, pp. 1443 – 51, 1995. [cited at p. 6, 104]

[13] B. Goode, "Voice over Internet Protocol (VoIP)," *Proceedings of the IEEE*, vol. 90, no. 9, pp. 1495 – 1517, 2002. [cited at p. 6, 139]

[14] R. Moraes, F. Vasques, and P. Portugal, "A forcing collision resolution approach able to prioritize real-time traffic in CSMA-based networks," *submitted to Computer Communications (ISSN 0140-3664)*. [cited at p. 6, 43]

[15] R. Moraes, F. Vasques, P. Portugal, and J. A. Fonseca, "A traffic separation mechanism (TSm) allowing the coexistence of CSMA and real-time traffic in wireless 802.11e networks," *WSEAS Transactions on Communications*, vol. 5, no. 5, pp. 890 – 897, 2006. [cited at p. 6, 43, 172]

[16] R. Moraes and F. Vasques, "Probabilistic timing analysis of the h-BEB collision resolution algorithm," *6th IFAC International Conference on Fieldbus Systems and their Applications (FET)*, pp. 107 – 114, 2005. [cited at p. 6, 43, 87]

[17] ——, "A probabilistic analysis of traffic separation in shared Ethernet systems using the h-BEB collision resolution algorithm," *13th International Conference on Real-Time Systems (RTS)*, pp. 74 – 96, 2005. [cited at p. 6, 43, 70]

[18] F. Carreiro, R. Moraes, J. A. Fonseca, and F. Vasques, "Real-time communication in unconstrained shared Ethernet networks: The virtual token-passing approach," *10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 425 – 432, 2005. [cited at p. 6, 75]

[19] R. Moraes, F. Vasques, P. Portugal, and J. A. Fonseca, "VTP-CSMA: A virtual token passing approach for real-time communication in IEEE 802.11 wireless networks," *To appear in IEEE Transactions on Industrial Informatics*, vol. 3, no. 3, August 2007. [cited at p. 7, 113]

[20] R. Moraes, F. Vasques, P. Portugal, and J. Fonseca, "Real-time communication in 802.11 networks: the virtual token passing VTP-CSMA approach," *31st IEEE Conference on Local Computer Networks (LCN)*, pp. 389 – 396, 2006. [cited at p. 7, 113]

[21] R. Moraes, P. Portugal, S. Vitturi, F. Vasques, and P. Souto, "Real-time communication in 802.11 networks: timing analysis and a ring management scheme for the VTP-CSMA architecture," *Submitted to 32nd IEEE Conference on Local Computer Networks (LCN)*, 2007. [cited at p. 7, 113]

[22] R. Moraes, P. Portugal, and F. Vasques, "Simulation analysis of the IEEE 802.11e EDCA protocol for an industrially-relevant real-time communication scenario," *11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 202 – 209, 2006. [cited at p. 7, 31, 97]

[23] ——, "A stochastic petri net model for the simulation analysis of the IEEE 802.11e EDCA communication protocol," *10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 38 – 45, 2006. [cited at p. 7, 126, 171]

[24] "IEEE Standard for Information Technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements. part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical

layer specifications," IEEE Std 802.3, 1998 Edition, 1998. [cited at p. 9, 12, 45]

[25] W.-H. Chen and C.-C. Lu, "CSMA/CD/TDMA: A dynamic combination for voice and data integration," *Proceedings - IEEE INFOCOM*, pp. 842 – 849, 1990. [cited at p. 12]

[26] D. Pritty, J. Malone, D. Smeed, S. Banerjee, and N. Lawrie, "Real-time upgrade for Ethernet based factory networking," *IECON Proceedings (Industrial Electronics Conference)*, vol. 2, pp. 1631 – 1637, 1995. [cited at p. 12]

[27] H. Kopetz and G. Bauer, "The time-triggered architecture," *Proceedings of the IEEE*, vol. 91, no. 1, pp. 112 – 126, 2003. [cited at p. 12]

[28] P. Pedreiras, P. Gai, L. Almeida, and G. C. Buttazzo, "FTT-Ethernet: A flexible real-time communication protocol that supports dynamic QoS management on Ethernet-based systems," *IEEE Transactions on Industrial Informatics*, vol. 1, no. 3, pp. 162 – 172, 2005. [cited at p. 13]

[29] P. B. R. Pedreiras, "Supporting flexible real-time communication on distributed systems," Ph.D. dissertation, University of Aveiro, 2003. [cited at p. 13]

[30] "Ethernet Powerlink protocol," Available at http://www.ethernet-powerlink.org. [cited at p. 13, 79]

[31] R. Grow, "A timed token protocol for local area networks," *Electro/82 Conference Record*, pp. 17 – 3, 1982. [cited at p. 14]

[32] C. Venkatramani and T.-C. Chiueh, "Supporting real-time traffic on Ethernet," *Real-Time Systems Symposium (RTSS)*, pp. 282 – 286, 1994. [cited at p. 14]

[33] J. M. Martinez and M. G. Harbour, "RT-EP: A fixed-priority real time communication protocol over standard Ethernet," *10th Ada-Europe International Conference on Reliable Software Technologies (Lecture Notes in Computer Science)*, vol. 3555, pp. 180 – 195, 2005. [cited at p. 15]

[34] J.-Y. Lee, H. Moon, S. Moon, K. W., S. Lee, and I. Park, "Token passing bus access method on the IEEE 802.3 physical layer for distributed control networks," *15th IFAC Workshop on Distributed Computer Control Systems*, 1998. [cited at p. 15]

[35] "IEEE Standard for Information Technology - "information Processing Systems - Local Area Networks - part 4: Token-passing bus access method and physical layer specifications"," ANSI/IEEE Std 802.4-1990 (Revision of ANSI/IEEE 802.4-1985), 1990. [cited at p. 15]

[36] "IEEE standard for local and metropolitan area networks media access control (MAC) bridges," IEEE Std 802.1D-2004 (Revision of IEEE Std 802.1D-1998), 2004. [cited at p. 15]

[37] "IEEE standards for local and metropolitan area networks: virtual bridged local area networks," IEEE Std 802.1Q, 1999. [cited at p. 15]

[38] J.-D. Decotignie, "A perspective on Ethernet-TCP/IP as a fieldbus," *4th International Conference on Fieldbus Systems and their Application*, 2002. [cited at p. 15]

[39] E. Jasperneite and P. Neumann, "Switched Ethernet for factory communication," *8th International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 205 – 212, 2001. [cited at p. 16, 78]

[40] P. Pedreiras and L. Almeida, *Approaches to enforce real-time behavior in Ethernet*, ser. The Industrial Information Technology Handbook, R. Zurawski. CRC Press LLC, 2005, pp. 20–1–20–28. [cited at p. 16, 40]

[41] S. Varadarajan and T.-C. Chiueh, "EtheReal: A host-transparent real-time fast Ethernet switch," *International Conference on Network Protocols (ICNP)*, pp. 12 – 21, 1998. [cited at p. 16]

[42] H. Hoang and M. Jonsson, "Switched real-time Ethernet in industrial applications: asymmetric deadline partitioning scheme," *2nd Internation Workshop on Real-Time LANs in the Internet Age*, 2003. [cited at p. 16]

[43] A. Takagi, S. Yamada, and S. Sugawara, "CSMA/CD with deterministic contention resolution." *IEEE Journal on Selected Areas in Communications*, vol. 1, no. 5, pp. 877 – 884, 1983. [cited at p. 16, 37]

[44] G. Le Lann and N. Rivierre, "Real-time communications over broadcast networks: the CSMA-DCR and the DOD-CSMA-CD protocols," *INRIA 1863*, 1993. [cited at p. 17]

[45] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard environment," *Journal ACM*, vol. 20, no. 1, pp. 46–61, 1973. [cited at p. 17]

[46] P. M. Gopal and J. W. Wong, "Analysis of a hybrid token-CSMA/CD protocol for bus networks." *Computer Networks and ISDN Systems*, vol. 9, no. 2, pp. 131 – 141, 1985. [cited at p. 17]

[47] J. L. Sobrinho and A. Krishnakumar, "EQuB - Ethernet Quality of Service using black bursts," *Conference on Local Computer Networks*, pp. 286 – 296, 1998. [cited at p. 18]

[48] R. Yavatkar, P. Pai, and R. Finkel, "Reservation-based CSMA protocol for integrated manufacturing networks," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 8, pp. 1247 – 1258, 1994. [cited at p. 19]

[49] M. L. Molle and L. Kleinrock, "Virtual time CSMA: Why two clocks are better than one." *IEEE Transactions on Communications*, vol. 33, no. 9, pp. 919 – 933, 1985. [cited at p. 19, 39]

[50] W. Zhao and K. Ramamritham, "Virtual time CSMA protocols for hard real-time communication." *IEEE Transactions on Software Engineering*, vol. SE-13, no. 8, pp. 938 – 952, 1987. [cited at p. 20, 21, 39]

[51] M. N. El-Derini and M. R. El-Sakka, "A CSMA protocol under a priority time constrained for real-time communication," *Second IEEE Workshop on Future Trends of Distributed Computing Systems Proceedings*, pp. 128 – 134, 1990. [cited at p. 21]

[52] M. L. Molle, "Prioritized-virtual-time CSMA: Head-of-the-line priority classes without added overhead," *IEEE Transactions on Communications*, vol. 39, no. 6, pp. 915 – 927, 1991. [cited at p. 21]

[53] R. Gallager, "Conflict resolution in random access broadcast networks," *AFOSR Workshop Communication Theory Applications*, pp. 74–76, 1978. [cited at p. 21, 39]

[54] J. F. Kurose, M. Schwartz, and Y. Yemini, "Multiple-access protocols and time-constrained communication." *Computing Surveys*, vol. 16, no. 1, pp. 43 – 70, 1984. [cited at p. 21]

[55] ——, "Controlling window protocols for time-constrained communication in multiple access networks." *IEEE Transactions on Communications*, vol. 36, no. 1, pp. 41 – 49, 1988. [cited at p. 21]

[56] R.-H. Jan and Y.-J. Yeh, "CSMA/CD protocol for time-constrained communication on bus networks," *IEE Proceedings, Part I: Communications, Speech and Vision*, vol. 140, no. 3, pp. 197 – 202, 1993. [cited at p. 22, 39]

[57] M. L. Molle, "A new binary logarithmic arbitration method for Ethernet," *Computers Research Institute, University of Toronto*, 1994. [cited at p. 22, 39, 66]

[58] B. Whetten, S. Steinberg, and D. Ferrari, "Packet starvation effect in CSMA/CD LANs and a solution," *Conference on Local Computer Networks*, pp. 206 – 217, 1994. [cited at p. 22]

[59] K. Ramakrishnan and H. Yang, "Ethernet capture effect: Analysis and solution," *Conference on Local Computer Networks*, pp. 228 – 240, 1994. [cited at p. 23, 39]

[60] S.-K. Kweon and K. G. Shin, "Statistical real-time communication over Ethernet," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 3, pp. 322–335, 2003. [cited at p. 23, 39]

[61] S.-K. Kweon and K. Shin, "Achieving real-time communication over Ethernet with adaptive traffic smoothing," *Proceedings Sixth IEEE Real-Time Technology and Applications Symposium*, pp. 90 – 100, 2000. [cited at p. 23, 24, 39, 40]

[62] M. Butto, E. Cavallero, and A. Tonietti, "Effectiveness of the 'leaky bucket' policing mechanism in ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 3, pp. 335 – 42, 1991. [cited at p. 24]

[63] L. Lo Bello, G. A. Kaczynski, and O. Mirabella, "Improving the real-time behavior of Ethernet networks using traffic smoothing," *IEEE Transactions on Industrial Informatics*, vol. 1, no. 3, pp. 151–161, 2005. [cited at p. 24, 34, 40]

[64] J. Son, I.-G. Lee, H.-J. Yoo, and S.-C. Park, "An effective polling scheme for IEEE 802.11e," *IEICE Transactions on Communications*, vol. E88-B, no. 12, pp. 4690 – 4693, 2005. [cited at p. 27]

[65] S.-C. Lo, G. Lee, and W.-T. Chen, "An efficient multipolling mechanism for IEEE 802.11 wireless LANs," *IEEE Transactions on Computers*, vol. 52, no. 6, pp. 764 – 68, 2003. [cited at p. 27]

[66] S. Lee, K. N. Ha, J. H. Park, and K. C. Lee, "NDIS-based virtual polling algorithm of IEEE 802.11b for guaranteeing the real-time requirements," *IECON Proceedings (Industrial Electronics Conference)*, pp. 2427 – 2432, 2005. [cited at p. 27]

[67] M. Ergen, D. Lee, R. Sengupta, and P. Varaiya, "WTRP-wireless token ring protocol," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 6, pp. 1863 – 1881, 2004. [cited at p. 28]

[68] A. Willig, "A MAC protocol and a scheduling approach as elements of a lower layers architecture in wireless industrial LANs," *IEEE International Workshop on Factory Communication Systems (WFCS'97)*, pp. 139 – 48, 1997. [cited at p. 28]

[69] D. Miorandi and S. Vitturi, "Analysis of master-slave protocols for real-time industrial communications over IEEE 802.11 WLANs," *2nd IEEE International Conference on Industrial Informatics*, pp. 143 – 148, 2004. [cited at p. 29, 37]

[70] F. De Pellegrini, D. Miorandi, S. Vitturi, and A. Zanella, "On the use of wireless networks at low level of factory automation systems," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 2, pp. 129 – 143, 2006. [cited at p. 29]

[71] R.-G. Cheng, C.-Y. Wang, L.-H. Liao, and J.-S. Yang, "Ripple: a wireless token-passing protocol for multi-hop wireless mesh networks," *IEEE Communications Letters*, vol. 10, no. 2, pp. 123 – 5, 2006. [cited at p. 29]

[72] J. Sobrinho and A. Krishnakumar, "Quality-of-service in ad hoc carrier sense multiple access wireless networks," *IEEE J. Sel. Areas Commun*, vol. 17, no. 8, pp. 1353 – 68, 1999. [cited at p. 29, 40]

[73] G.-H. Hwang and D.-H. Cho, "New access scheme for VoIP packets in IEEE 802.11e wireless LANs," *IEEE Communications Letters*, vol. 9, no. 7, pp. 667 – 669, 2005. [cited at p. 30]

[74] J.-P. Sheu, C.-H. Liu, S.-L. Wu, and Y.-C. Tseng, "A priority MAC protocol to support real-time traffic in ad hoc networks," *Wireless Networks*, vol. 10, no. 1, pp. 61 – 69, 2004. [cited at p. 31]

[75] A. Hamidian and U. Korner, "An enhancement to the IEEE 802.11e EDCA providing QoS guarantees," *Telecommunication Systems*, vol. 31, no. 2-3, pp. 195 – 212, 2006. [cited at p. 32]

[76] "IEEE P802.11e/D10.0, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 7: Medium Access Control (MAC) Quality of Service Enhancements," September 2005. [cited at p. 32]

[77] C. Wang, B. Li, and L. Li, "A new collision resolution mechanism to enhance the performance of IEEE 802.11 DCF," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 4, pp. 1235 – 1246, 2004. [cited at p. 33]

[78] X. Yang and N. Vaidya, "Priority scheduling in wireless ad hoc networks," *Wireless Networks*, vol. 12, no. 3, pp. 273 – 286, 2006. [cited at p. 33]

[79] N. Vaidya, A. Dugar, S. Gupta, and P. Bahl, "Distributed fair scheduling in a wireless LAN," *IEEE Trans. Mob. Comput*, vol. 4, no. 6, pp. 616 – 29, 2005. [cited at p. 34]

[80] S. Golestani, "A self-clocked fair queueing scheme for broadband applications," *Proceedings IEEE INFOCOM '94. The Conference on Computer Communications. Networking for Global Communications*, vol. 2, pp. 636 – 46, 1994. [cited at p. 34]

[81] E. Lopez-Aguilera, J. Casademont, J. Cotrina, and A. Rojas, "Enhancement proposal for WLAN IEEE 802.11e: Desynchronization of its working procedure," *14th IEEE Workshop on Local and Metropolitan Area Networks LANMAN*, pp. 1–6, 2005. [cited at p. 34]

[82] L. Lo Bello, F. S. Kaczynski, and O. Mirabella, "A wireless traffic smoother for soft real-time communications over IEEE 802.11 industrial networks," *11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1073–1079, 2006. [cited at p. 34]

[83] "Spectralink voice priority - quality of service for voice traffic in wireless LANs," SpectraLink Wireless@Work, available at www.spectralink.com. [cited at p. 34]

[84] G.-H. Hwang and D.-H. Cho, "Fast retransmission mechanism for VoIP in IEEE 802.11e wireless LANs," *IEEE 60th Vehicular Technology Conference (VTC2004-Fall)*, vol. 7, pp. 4996 – 5000, 2004. [cited at p. 34, 35]

[85] C. Casetti, C.-F. Chiasserini, M. Fiore, and M. Garetto, "Notes on the inefficiency on 802.11e HCCA," Dipartimento de Elettronica, Politecnico di Torino, 2005. [cited at p. 37]

[86] P. Garg, R. Doshi, R. Greene, M. Baker, M. Malek, and X. Cheng, "Using IEEE 802.11e MAC for QoS over wireless," *Conference Proceedings of the 2003 IEEE International Performance, Computing, and Communications Conference*, pp. 537 – 42, 2003. [cited at p. 37, 190]

[87] J. R. Moyne and D. M. Tilbury, "The emergence of industrial control networks for manufacturing control, diagnostics, and safety data," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 29–47, 2007. [cited at p. 40]

[88] R. Moraes and F. Vasques, "Real-time traffic separation in shared Ethernet networks: simulation analysis of the h-BEB collision resolution algorithm," *11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pp. 89 – 92, 2005. [cited at p. 43, 88]

[89] "IEEE Standard for Information Technology - "Logical Link Control"," ANSI/IEEE 802.2 Std, 1998 edition (Incorporating ANSI/IEEE Stds 802.2c-1997, 802.2f-1997), 1998. [cited at p. 45, 114]

[90] K. J. Christensen, "Performance evaluation of the binary logarithmic arbitration method (BLAM)," *Conference on Local Computer Networks (LCN)*, pp. 396 – 403, 1996. [cited at p. 47]

[91] J. Deng and R.-S. Chang, "A priority scheme for IEEE 802.11 DCF access method," *IEICE Trans. Commun. (Japan)*, vol. E82-B, no. 1, pp. 96 – 102, 1999. [cited at p. 48]

[92] "NS-2 network simulator - version 2.26," Available at http://www.isi.edu/nsnam/ns, 2006. [cited at p. 56, 172]

[93] Q. Ni, L. Romdhani, and T. Turletti, "A survey of QoS enhancements for IEEE 802.11 wireless LAN," *Wireless Communications and Mobile Computing*, vol. 4, no. 5, pp. 547 – 566, 2004. [cited at p. 56, 106, 190]

[94] K. J. Christensen, "Simulation study of enhanced arbitration methods for improving Ethernet performance," *Computer Communications*, vol. 21, no. 1, pp. 24 – 36, 1998. [cited at p. 60, 62]

[95] D. R. Boggs, J. C. Mogul, and C. A. Kent, "Measured capacity of an Ethernet: myths and reality," *Computer Communication Review*, vol. 25, no. 1, pp. 123 – 136, 1995. [cited at p. 62, 66]

[96] J. Hastad, T. Leighton, and B. Rogoff, "Analysis of backoff protocols for multiple access channels." *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 241–253, 1987. [cited at p. 66]

[97] S. S. Lam, "Carrier sense multiple access protocol for local networks." *Computer Networks: The International Journal of Distributed Informatique*, vol. 4, no. 1, pp. 21–32, 1980. [cited at p. 66]

[98] Y.-C. Liu and G. L. Wise, "Performance of a CSMA/CD protocol for local area networks." *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, no. 6, pp. 948 – 955, 1987. [cited at p. 66]

[99] R. M. Metcalfe and D. R. Boggs, "Ethernet: distributed packet switching for local computer networks." *Communications of the ACM*, vol. 19, no. 7, pp. 395 – 404, 1976. [cited at p. 66, 67, 68]

[100] S. S. Lam and L. Kleinrock, "Packet switching in a multiaccess broadcast channel: dynamic control procedures." *IEEE Transactions on Communications*, vol. CM-23, no. 9, pp. 891 – 904, 1975. [cited at p. 67]

[101] G. T. Almes and E. D. Lazowska, "Behavior of Ethernet-like computer communications networks." *AIChE Symposium Series*, pp. 66 – 81, 1979. [cited at p. 67]

[102] J. Feld, "Real-time communication in Profinet v2 and v3 designed for industrial purposes," *5th IFAC International Conference on Fieldbus Systems and their applications*, pp. 291–296, 2003. [cited at p. 77]

[103] A. Boller, "Profinet v3: Bringing hard real-time and the IT world together," http://www.controleng.com/article/CA318939.html, September 2003. [cited at p. 77]

[104] P. Brooks, "Ethernet/IP - Industrial protocol," *8th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 505 – 514, 2001. [cited at p. 77]

[105] "IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems," IEEE Std 1588-2002, 2002. [cited at p. 77]

[106] D. Jansen and H. Buttner, "Real-time Ethernet the EtherCAT solution," *Computing & Control Engineering Journal*, vol. 15, no. 1, pp. 16–21, 2004. [cited at p. 77]

[107] J. Feld, "Profinet - scalable factory communication for all applications," *IEEE International Workshop on Factory Communication Systems (WFCS)*, pp. 33 – 38, 2004. [cited at p. 78]

[108] T. Sauter, *Fieldbus Systems: History and Evolution*, ser. The Industrial Information Technology Handbook, R. Zurawski. CRC Press LLC, 2005, pp. 7–1–7–39. [cited at p. 79]

[109] C. Spurgeon, *Ethernet: The definitive Guide*, M. Stone and C. Toporek, Eds. Sebastopol: O´Reilly & Associates, Inc., 2000. [cited at p. 84, 87]

[110] J. A. Fonseca, P. Bartolomeu, V. Silva, and F. Carreiro, "On the practical issues of implementing the VTPE-hBEB protocol in small processing power controllers," *Submitted to 12th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2007. [cited at p. 90]

[111] "Cyrrus logic," Available at: http://www.cirrus.com/en/, 2007. [cited at p. 90]

[112] J. Ayres, "Using the crystal CS8900A in 8-bit mode - Cirrus application note AN181," Available at http://www.cirrus.com/en/. [cited at p. 90]

[113] "Microchip," Available at: http://microchip.com, 2007. [cited at p. 90]

[114] "Silabs," Available at: http://www.silabs.com, 2007. [cited at p. 90]

[115] "Edtp," Available at: http://www.edtp.com, 2007. [cited at p. 90]

[116] "D-ITG,Distributed Internet Traffic Generator," Available at: http://www.grid.unina.it/software/ITG/, 2007. [cited at p. 92]

[117] A. Willig and A. Wolisz, "Ring stability of the PROFIBUS token-passing protocol over error-prone links," *IEEE Transactions on Industrial Electronics*, vol. 48, no. 5, pp. 1025 – 1033, 2001. [cited at p. 100, 126, 183]

[118] "IEEE standard for information technology - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: high-speed physical layer in the 5ghz band." IEEE Std 802.11a-1999, 1999. [cited at p. 101]

[119] Q. Ni, "Performance analysis and enhancements for IEEE 802.11e wireless networks," *IEEE Network*, vol. 19, no. 4, pp. 21 – 27, 2005. [cited at p. 106]

[120] J. F. Kurose and R. K. W., *Computer Networking, a Top-Down Approach Featuring the Internet*, 3rd ed. Addison-Welsley Longman, 2004. [cited at p. 108]

[121] H. Zhai, X. Chen, and Y. Fang, "How well can the IEEE 802.11 wireless LAN support quality of service?" *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 3084 – 94, 2005. [cited at p. 124]

[122] G. Agrawal, B. Chen, and S. Davari, "Guaranteeing synchronous message deadlines with the timed token medium access control protocol," *IEEE Transactions on Computers*, vol. 43, no. 3, pp. 327 – 339, 1994. [cited at p. 126, 152]

[123] J.-P. Thomesse, "Fieldbus technology in industrial automation," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1073 – 1101, 2005. [cited at p. 128, 129]

[124] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535 – 47, 2000. [cited at p. 171]

[125] P. E. Engelstad and O. N. Osterbo, "Non-saturation and saturation analysis of IEEE 802.11e EDCA with starvation prediction," *Proceedings of the Eighth ACM Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 224 – 233, 2006. [cited at p. 171]

[126] J.-D. Kim and C.-K. Kim, "Performance analysis and evaluation of IEEE 802.11e EDCF," *Wireless Communications and Mobile Computing*, vol. 4, no. 1, pp. 55 – 74, 2004. [cited at p. 171]

[127] Z.-N. Kong, D. H. K. Tsang, B. Bensaou, and D. Gao, "Performance analysis of IEEE 802.11e contention-based channel access," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 10, pp. 2095 – 2106, 2004. [cited at p. 171]

[128] P. P. Pham, "Comprehensive analysis of the IEEE 802.11," *Mobile Networks and Applications*, vol. 10, no. 5, pp. 691 – 703, 2005. [cited at p. 171]

[129] J. W. T. Robinson, "An analytical model for the service delay distribution of IEEE 802.11e enhanced distributed channel access," Master's thesis, School of Engeneering Science, 2005. [cited at p. 171, 172, 185, 186, 187]

[130] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: Throughput and delay analysis," *Computer Communications*, vol. 25, no. 3, pp. 313 – 321, 2002. [cited at p. 171]

[131] "O. Tech. OPNET," Available at: http://www.opnet.com, 2006. [cited at p. 172]

[132] R. German and A. Heindl, "Performance evaluation of IEEE 802.11 wireless LANs with stochastic Petri nets," *Proceedings 8th International Workshop on Petri Nets and Performance Models*, pp. 44 – 53, 1999. [cited at p. 172, 173]

[133] A. Heindl and R. German, "Performance modeling of IEEE 802.11 wireless LANs with stochastic Petri nets," *Performance Evaluation*, vol. 44, no. 1-4, pp. 139 – 64, 2001. [cited at p. 172, 173]

[134] ——, "The impact of backoff, EIFS, and beacons on the performance of IEEE 802.11 wireless LANs," *Proceedings IEEE International Computer Performance and Dependability Symposium (IPDS)*, pp. 103 – 12, 2000. [cited at p. 172, 173]

[135] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541 – 580, 1989. [cited at p. 173]

[136] R. M. Z. Zurawski, "Petri nets and industrial applications: A tutorial," *IEEE Transactions on Industrial Electronics*, vol. 41, no. 6, pp. 567 – 583, 1994. [cited at p. 173]

[137] J. Billington, M. Diaz, and G. Rozenberg, *Application of Petri nets to communication networks: Advances in Petri nets*, ser. Lecture Notes in Computer Science. Berlim: Springer-Verlag, 1999. [cited at p. 173]

[138] R. German, *Performance Analysis of Communication Systems - Modeling with Non-Markovian Stochastic Petri Nets*, ser. Wiley-interscience series in systems and optimization. Chichester: John Wiley & Sons, 2000. [cited at p. 173, 189]

[139] P. Portugal, A. Carvalho, and F. Vasques, "A model based on a stochastic Petri net approach for dependability evaluation of controller area networks (CAN)," *5th IFAC International Conference on Fieldbus Systems and their Applications*, 2005. [cited at p. 173]

[140] W. Sanders and J. Meyer, *Stochastic Activity Networks: Formal Definitions and Concepts.* Lectures Notes in Computer Science, pp. 315–343. [cited at p. 173, 174]

[141] C. Lindeman, *Performance Modelling with Deterministic and Stochastic Petri nets.* Chichester: John Wiley & Sons, 1998. [cited at p. 173, 174, 189]

[142] W. H. Sanders, "Möbius user manual, University of Illinois." [cited at p. 173, 174]

[143] L. Kleinrock, *Queuing Systems.* New York: John Wiley & Sons, 1975, vol. 1&2. [cited at p. 189]

[144] S. Mangold, C. Sunghyun, O. Klein, G. Hiertz, and L. Stibor, "IEEE 802.11 wireless LAN for quality of service," *European Wireless 2002*, vol. 1, pp. 32–39, 2002. [cited at p. 190, 191]

[145] S. Wiethoelter, M. Emmelmann, C. Hoene, and A. Wolisz, "TKN EDCA model for NS-2 (TKN-06-003)," Technical University of Berlin - Telecommunication Networks Group, Tech. Rep., 2006. [cited at p. 190]

[146] S. Wiethoelter and C. Hoene, "Design and verification of an IEEE 802.11e EDCF simulation model in NS-2.26 (TKN-03-19)," Technical

University Berlin - Telecommunication Networks Group, Tech. Rep., 2003. [cited at p. 190]

Appendices

Appendix A

A SPN model of the EDCA mechanism

In this Appendix, it is described a simulation model that enables the evaluation of IEEE 802.11e EDCA networks, focusing in the behavior of the Enhanced Distributed Channel Access (EDCA) function associated to Quality of Service (QoS) stations. This appendix is largely drawn from the following published work: "A Stochastic Petri Net Model for the Simulation Analysis of the IEEE 802.11e EDCA Communication Protocol" (Moraes et al. [23]).

A.1 Introduction

Traditionally, performance analysis of IEEE 802.11e communication networks has been carried out by developing evaluation models from two distinct points of view: analytical and simulation. Analytical models [124, 125, 126, 127, 128, 129, 130] have the advantage to provide analytical expressions/formalisms that helps to analyze the influence of different parameters. Besides, they also (usually) provide quick results. However, this type of solution typically compels to the adoption of simplistic assumptions. Either the protocol behavior is simplified or specific load scenarios are assumed (*e.g.* saturation). Moreover, the characteristics of the analytical models usually limit both the nature and the number of performance measures that can be obtained.

For more realistic scenarios, simulation techniques are necessary. Several simulation analysis have been done using the *Network Simulator (NS-2)* [92] or the OPNET tools [131]. The NS-2 tool is an open source discrete event simulator, whereas the OPNET tool has been developed by OPNET Technologies, Inc. Both are specially suited to analyze the performance of communication networks. In one of ours previous works [15], the NS-2 tool was initially selected to simulate the IEEE 802.11e behavior when supporting real-time communications. However, many doubts emerged about the model validation, as it is difficult to understand how both the protocol and its timing characteristics are modeled. Furthermore, some inconsistencies of the NS-2 model have been identified and explained by [129], which increased the doubts about the validity of the obtained results. Also, from our previous experience, it demonstrated to be hard to implement any slight protocol modification, or even to modify its timing characteristics, requiring many hours of implementation work.

Related Work: There are few available papers that use SPNs as a modeling formalism to analyze the IEEE 802.11 communication protocols [132, 133, 134]. However, to our best knowledge, the model presented here is the first SPN model that covers IEEE 802.11e EDCA.

In [132], it is proposed both simulation and analytical SPN models to evaluate the performance of the IEEE 802.11. These models describe the minimal behavior of the Distributed Coordination Function (DCF): Basic Access based on a two-way handshaking and Request-to-Send/Clear-to-Send (RTS/CTS) based on a four-way handshaking. The simulation model has the required detail to describe the main features of the protocol, while the analytical one is much more compact and simpler in order to obtain an analytically feasible solution. Both models assume ideal channel conditions and do not consider either retransmissions or Extended Interframe Spaces (EIFS). Besides, they include some inaccuracies/simplifications, particularly in the manner how backoff procedures and timeouts are modeled.

In [133, 134], the previous simulation model is extended to incorporate other aspects such Beacon frames and EIFS. The analytical model maintains the same characteristics, but with a better definition of some parameters. However, this new models suffers from the same problems as the former ones.

Although from a modeling point of view the previous models establish important contributions, their implementation in most of the SPNs tools cannot be done without some difficulties. It occurs due to the absence, in

those tools, of a formalism that helps to build automatically several replicas of the model. This is essential in the evaluation of scenarios composed by several stations (models). Otherwise, it is necessary to replicate manually each model, which can be an error-prone and time-consuming task.

In this appendix, we present the most important implementation details about a SPN simulation model that comprises a precise and detailed implementation of the Enhanced Distributed Channel Access (EDCA) function associated to Quality of Service (QoS) stations, considering both their functional and temporal perspectives.

A.2 Stochastic Petri Nets

Petri Nets (PN) are a graphical and mathematical modeling tool which enables the description and analysis of dynamic systems where concurrent, asynchronous, distributed, parallel, timed, non-deterministic, and stochastic activities are present [135]. These properties characterize discrete-event systems (DEDS's), whose examples include industrial automated systems, communication systems and computer-based systems [136]. Over the last decade Stochastic Petri Nets (SPNs) have become a widely used framework for performance and dependability evaluation of various kinds of systems by several reasons [137, 138, 132, 133, 134, 135, 139, 140, 136, 141, 142]:

- A graphical and intuitive description of the system behavior, which can be used as a visual-communication aid between different users;
- Representation of complex systems by very compact models using a non-ambiguous and simple notation;
- A formal basis, where it is possible to set up algebraic equations and other mathematical models (*e.g.* stochastic processes) reflecting the dynamics of the system;
- The opportunity to get different types of solutions using the same model;
- Independence between the developed model and the support tool used in the analysis/solution, with multiple available support tools.

The use of SPNs to obtain performance models can be performed from two viewpoints, according to the type of required solution: *Analytical*, when the SPNs obey to certain structural rules and algebraic/analytical process can be automatically generated and solved; *Simulation*, in this case neither of the previous limitations are present and a significant number of SPNs modeling extensions are available to reduce the model complexity. Since the SPNs semantics are formally well-established, models are easily constructed and less error-prone than custom simulation programs.

A.2.1 Brief Description of the Modeling Tool

The proposed model was implemented using the Möbius tool [142], which supports an SPN extension, referred as Stochastic Activity Networks (SANs) [140]. It provides an hierarchical modeling approach that is combined with state-of-the-art analytical and simulation solutions. The modeling formalism is quite similar to the classic SPNs with four primitive objects: *places*, *activities*, *input gates* and *output gates* (Figure A.1). The interaction (data flow) between these objects is described by means of *arcs*. Further details can be found at [140, 142].

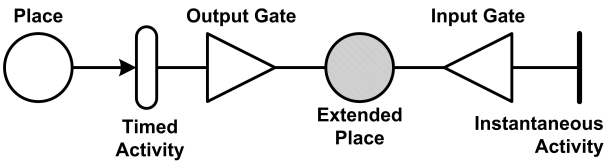

Figure A.1: SANs primitive objects.

Places represent system resources or the state of the modeled system. They are represented graphically as circles. Each place contains a certain number of *tokens*, which represents the *marking* of the place. There are two types of places: *standard* and *extended*. Standard places are similar to PN places and always contain an integer number of tokens. Extended places implement a formalism which is analogous to Colored Petri Nets [141]. In this case, the marking of the place is defined by means of a data structure (*e.g.* C++ structure).

Activities are similar to PN transitions and represent actions in the modeled system that take some specific amount of time to complete. They are graphically represented as rectangles (*timed* activities) or thin lines (*instantaneous* activities). When the activity *completes/fires* (*i.e.* the time

associated elapses) it moves tokens between places connected to the activity, reflecting a changing in the system state.

Input gates and *output gates* control the enabling of activities and define the marking changes that will occur when an activity completes. They are represented as oriented triangles. This behavior is implemented by means of C++ code.

One of the most interesting features of this tool is the possibility of building composed models. That is, the complete SAN model can be obtained through the combination of individual SAN models into a *composed-model*, using the following two constructs: *Rep* for defining replication of SANs and *Join* for combining several SANs. Both *Rep* and *Join* require that the constituent models have common-places (shared places), to provide communication among individual models.

A.3 Model Description

A.3.1 Modeling Strategy

To avoid the process of building a model from the scratch for each simulation scenario, it was developed a single *station model*. This model is later replicated, using the *Rep* primitive, to obtain the required network simulation scenario (Figure A.2, left). The number of replicas is parameterized by the user and fully automated by the tool. This provides an important flexibility in the evaluation process, as it speeds-up the analysis of different network scenarios. Besides, by using the *Join* primitive it is possible to evaluate scenarios where different types of stations can coexist in the same network (Figure A.2, right).

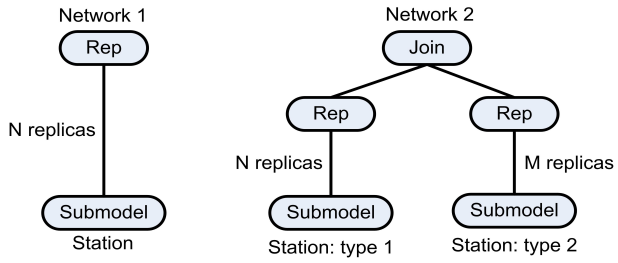

Figure A.2: Use of Rep and Join primitives to build different network scenarios.

Therefore, in a network scenario with N stations, there will be N station models (replicas) being simultaneously executed. Moreover, these models are independent and are not necessarily synchronized. Communication between models is performed by means of common-places.

A.3.2 Model Presentation

Due to the intensive use of C++ code (input and output gates) the model is described from a functional perspective (near to the SPN marking evolution) and without any mention to the internal code. The complete documentation about the model is available at: *http://www.fe.up.pt/∼vasques/ieee80211e/*.

To improve the clearness of the presentation, the model is split into three sub-models: **frame queuing**, **processing** and **exchange sequence**. Although each sub-model is separately represented, there is a close interaction between them. Additionally, only arcs that correspond to the most important data flows are represented and all aspects related with model initialization and some housekeeping procedures are not either represented or discussed.

Since each QoS station comprises four EDCA functions (each associated to an AC category) with the same functional behavior, it was used the concept of *extended places* to keep in a single place similar data about the four AC categories. These extended places are implemented as an array of 4 elements. Therefore, the index of this array, $i = \{0, 1, 2, 3\}$, contains the marking associated to i-th AC category (ACi).

The following terminology was used in the description of the model. Places whose label begins with a capital letter are common-places. Otherwise they are internal places (their marking is different among replicas). Places whose designation begins with the letters ac_ refers to *extended places*. Their marking is an array and each individual element, ac_(i), contains the marking associated to ACi. To simplify the discussion we also refer ac_(i) as a place.

A.3.3 Frame Queuing

This sub-model implements the arrival of frames from higher protocol layers to the queues associated to each AC category and also their management

(Figure A.3). Since all 4 queues share the same functional behavior, the discussion focuses in a generic one.

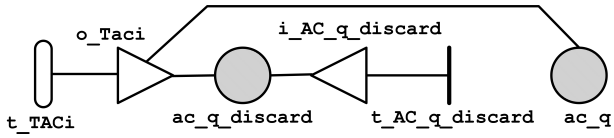

Figure A.3: Frame queuing sub-model.

The timed activity t_TACi models the inter-arrival time of frames to the queue ACi. The inter-arrival time represents the traffic generation pattern and is defined by a distribution function, which is fully configurable by the user. Typical functions employed are: exponential to model Poisson traffic, deterministic to model periodic traffic and erlang or normal to model periodic traffic with jitter.

When the activity t_TACi completes, a new frame arrives to queue ACi. The four AC queues are represented by the extended place ac_q, whose marking, ac_q(i), indicates the number of frames waiting for transmission in queue ACi. If the queue is not full, the frame waits in the queue for later transmission. Otherwise is discarded. In the former case the marking of ac_q(i) is incremented by one. In the latter, the extended place ac_q_discard(i) is marked with one token. The previous procedures are implemented by the output gate o_Taci.

Input gate i_AC_q_discard tests if there is a token in ac_q_discard(i), which, if is true, enables the immediate activity t_AC_q_discard. If this activity fires, the token is removed from ac_q_discard(i). This is only a modeling artifact that will enable later the evaluation of the number of frames discarded from each individual queue.

A.3.4 Frame Processing

This sub-model implements the Enhanced Distributed Channel Access (EDCA) function associated to each access category (AC). This is the main function implemented by QoS stations.

After the medium becomes idle and on specific *slot boundaries* each EDCA function in a QoS station shall perform one, and only one, of the following functions [8]:

- Initiate the transmission of a frame exchange sequence for that access function;
- Decrement the backoff counter for that access function;
- Invoke the backoff procedure due to an internal collision;
- Do nothing for that access function.

The slot boundaries (SB_n) of different EDCA function in the same QoS station always occur in the same instant and are multiples of *aSlotTime* (Figure A.4).

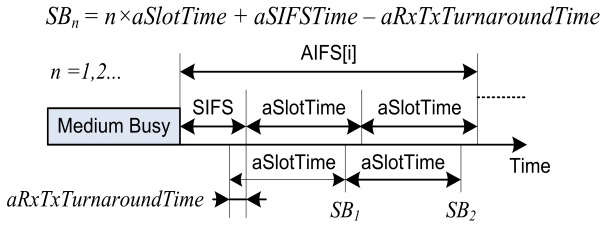

Figure A.4: Slot boundary definition.

Using the previous relationships, the frame processing sub-model was developed as follows (Figure A.5). After the medium becomes idle, a station waits during a SIFS interval (*aSIFSTime*). At the end of this interval, a *slot counting* procedure begins. This procedure counts *aSlotTime* intervals of idle medium. The end of these intervals corresponds to a slot boundary instant (SB_n), where the EDCA function is executed.

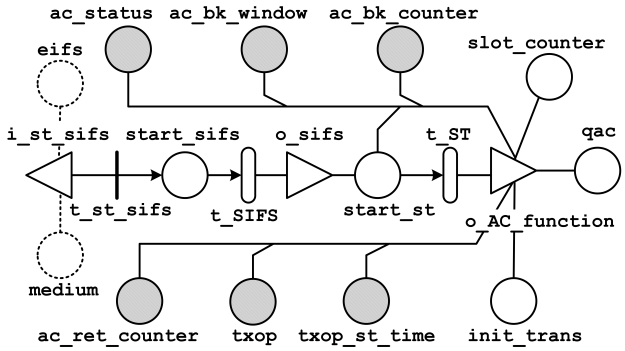

Figure A.5: Frame processing sub-model.

The input gate i_st_sifs verifies the necessary conditions that start a SIFS interval. Among these conditions are: haven't initiated the *slot counting*

procedure, the physical medium is idle, the current station is not transmitting (sending a data frame or waiting for an acknowledgement frame) and an eventual EIFS (Extended Interframe Space) interval elapsed. Places medium, qac and eifs represent, respectively, the physical medium status, the current transmitting AC queue and the elapsing of an EIFS interval. The behavior of these places will be discussed later.

When the previous conditions are fulfilled, the activity t_st_sifs fires and the place start_sifs is marked, which enables the timed activity t_SIFS. In order to guarantee synchronization with the slot boundaries (SB_n), the duration of this activity is defined as *aSIFSTime-aRxTxTurnaroundTime*.

When the activity t_SIFS completes, the place start_st is marked with one token if the medium is free. Otherwise, it is unmarked and all queues status goes to DEFER (discussed later). These actions are executed by the output gate o_sifs. When the place start_st is marked, the *slot counting* procedure begins by enabling the timed activity t_ST. This activity has a duration of *aSlotTime*.

In order to make a decision about which type of function must be executed, each queue maintains data about its status. Queue statuses are stored in the extended place ac_status by means of an identifier (a predefined number of tokens). The EDCA function is implemented by the execution of the output gate o_AC_function, when the activity t_ST completes. This gate implements a complex algorithm, which is (partially) represented in Figure A.6.

In simple terms the algorithm works as follows. When the activity t_ST completes, it removes the token from place start_st and, if the medium is idle, it increments the marking of slot_counter by one. This place is used to count the number of *idle medium* slots since the last slot boundary. From this counting, and using the queue status, it is possible to take a decision about which function to perform. A state-machine with the following states was used to define the marking of place ac_status(i):

- IMMEDIATE_ACCESS: an AIFS[i] time interval of idle medium had occurred and the queue is empty. As this time interval is related with the slot boundaries instants (Figure A.4), their occurrence can be inferred from the marking of slot_counter;
- DEFER: deferring the access. The medium is busy or an AIFS[i] time interval has still not elapsed;

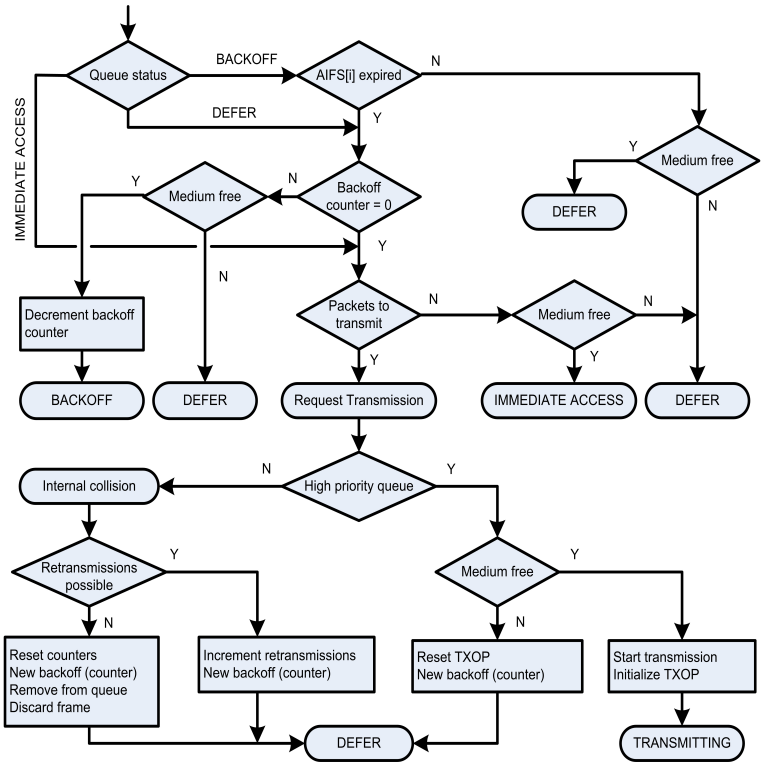

Figure A.6: Control Flow Summary - algorithm.

- BACKOFF: decrementing the backoff counter if the medium is idle;
- TRANSMITTING: transmitting a data frame or waiting for an acknowledge frame.

During the execution of o_AC_function there are other places which have their contents changed or used to perform several types of decisions. Extended place ac_bk_counter(i) is used to count the number of remaining backoff slots associated to queue ACi. Their marking is decremented by one during the *slot counting* procedure if the queue status is in BACKOFF. When it reaches at 0 the backoff counter has expired.

Extended place ac_bk_window(i) is used to maintain the current size of the backoff window associated to queue ACi. For each queue this window is updated after a frame exchange sequence or due to a collision (internal or external) according to pre-defined rules [8]. Extended place ac_ret_counter(i) is used to count the number of retransmissions of the data frame currently trying to be transmitted associated to queue ACi. The marking of this

place is incremented by one if there is a collision (internal or external). If a maximum is reached, the data frame is discarded and this counter is reset.

If a decision to transmit a data frame is performed (initiate a frame exchange sequence), the following places are affected: qac, init_trans, txop and txop_st_time. Place qac is used to indicate which queue is currently transmitting, and is only a model artifact. Place init_trans is used to indicate the beginning of a frame exchange sequence (see §A.3.5). In this case the status of the transmitting queue is changed to TRANSMITTING, while the others queues go to DEFER.

Places txop and txop_st_time are related with the implementation of the contention-based Transmission Opportunity (EDCA TXOP). To implement this mechanism is necessary to have a timer (an activity) that implements the TXOP interval and also the possibility to access anytime its current value. Unfortunately PNs do not support the possibility to access to the current time value of a previously enabled transition (activity). To overcome this problem it was used an internal function made available by Möbius that enables the access of the current simulation time. When this function is called from an output gate attached to an activity, it will return the firing instant of this activity, *i.e.* the current simulation time. Therefore, when the data frame begins their transmission the place txop is initialized with the TXOP default value associated to the queue and place txop_st_time is initialized with the current simulation time. So, the current value of TXOP can be obtained anytime by subtracting the marking of the previous places from the current simulation time. This is a modeling artifact that implements the TXOP timer in an indirect way.

If during the execution of o_AC_function the medium is considered busy then, among other actions, place start_st isn't marked, the marking of place slot_counter is reset and all the queues have its status changed to DEFER. This is equivalent to stop and reset the *slot counting* procedure, since the activity t_ST becomes disabled. If the medium is idle and if there isn't a data frame to transmit, the place start_st is marked again.

A.3.5 Frame Exchange Sequence

This sub-model implements the frame exchange sequence procedures (transmit a data frame or wait for an acknowledge frame) and the management of external collisions and interferences. In order to simplify the presentation, it is restricted the discussion to the Basic Access (two-way handshake). This

results from the fact that RTS/CTS implementation being similar from a modeling perspective. Besides, this sub-model is also split in several subnets for clearness of the presentation.

Collisions can only occur in a specific time interval defined as the *vulnerable period*. It begins in the instant where a station decides to transmit a data frame (a slot boundary) and it ends immediately before a slot boundary in the remote stations. This time interval consists of the air propagation time ($aAirPropagationTime$), *i.e.*, the time that takes to change from the receiving to the transmitting state ($aRxTxTurnaroundTime$), and the time whose receiver requires to access the medium within every slot time ($aCCAtime$). Since the later one is already indirectly included in the slot boundary definition [8, 7], it is only necessary to consider the former two.

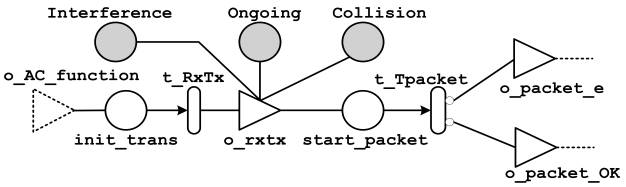

Figure A.7: Vulnerable period subnet.

Place init_trans is marked (§A.3.4) when the station takes the decision to transmit a data frame (Figure A.7). From that instant (a slot boundary) there is a minimum time that a station requires to change from the receiving to the transmitting state ($aRxTxTurnaroundTime$). This delay is represented by the timed activity t_RxTx. When this activity completes, the token is removed from place init_trans and the output gate o_rxtx is executed. This corresponds to the beginning of the transmission in the medium. When gate o_rxtx is executed, the contents of the following places are changed or tested: Ongoing, Collision and Interference. All of these are *common-places*.

Place Ongoing is used to represent the number of ongoing transmissions in the physical medium. Their marking is incremented by one when a data frame transmission begins. Note that their marking is updated by each replica (station models) initiating a transmission.

Place Collision is used to indicate if there is an external collision between data frames simultaneously transmitted by different stations. This is implemented in the following way. Before updating Ongoing, it is tested if their marking is >0. If is affirmative, then there is more than one data frame

being simultaneously transmitted, which is equivalent to say that there is a collision between data frames. In this case Collision is marked with one token. Otherwise, it is marked with 0 tokens. This is a model artifact, since the station knows only by indirect means that a collision had occurred.

Place Interference is used to indicate the existence of external interferences (*e.g.* EMI) during frame transmissions. For modeling the error characteristics of a wireless channel, it has been used a Gilbert-Elliot error model, where the channel is always in one of two states: Good or Bad (Figure A.8). This model assumes that bit errors are independent, with a fixed error rate in each state.

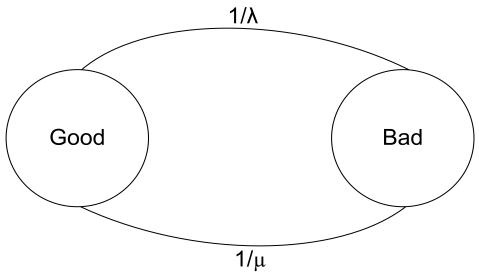

Figure A.8: A two state Markov channel.

For parametrization of the error model four value are necessary: BER (bit error rate) in good state e_g, BER in bad state e_b ($e_g << e_b$), mean duration of good state λ in seconds, mean duration of bad state μ in seconds. With $p_g = \lambda/(\lambda+\mu)$ and $p_b = \mu/(\lambda+\mu)$ being the steady-state probabilities for being in state good or bad. Therefore, the mean BER m is given by $m = p_g \times e_g + p_b \times e_b$. This model can also supporting a Semi-Markov error modeling as described in [117].

The *medium propagation subnet* (Figure A.9) is used anytime when a station (model) wants to inform the remaining replicas of a modification in the medium status. To perform this task two places are used: st_prop and medium. Place medium always reflects the physical medium status where the station is geographically localized. Their marking can take 2 values: IDLE and BUSY representing, respectively, an idle and busy medium. When a station wants to update or to change the medium status, it marks both medium and st_prop with the intended value. When st_prop is marked, the activity t_prop is enabled. This timed activity represents the air propagation time (*aAirPropagationTime*) that a frame takes to reach the remaining

stations. When this activity completes, the marking of st_prop is moved to the common-place R_medium.

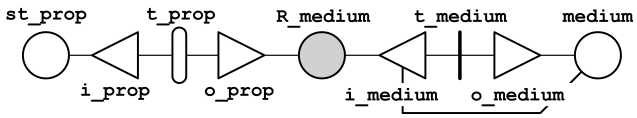

Figure A.9: Medium propagation subnet.

This R_medium place is used to trigger a mechanism that updates the marking of place medium in the remaining replicas. When R_medium is marked, the immediate activity t_medium is enabled in each replica (model) if the marking of R_medium is different from medium. Otherwise is disabled. When this activity completes, it copies the marking of R_medium to medium. In that way all the remaining replicas will have the place medium updated after *aAirPropagationTime*. Note that t_medium fires at maximum only once in each replica.

If a remote station is in the *slot counting* procedure (§A.3.4) and a BUSY medium status "arrives" before t_ST completes, then all queues in this station go to DEFER and the slot counting process is stopped and reset. However, if it "arrives" immediately after the instant when the station decides to begin a data frame transmission (slot boundary), then it has no effects. Note that this time interval corresponds to the vulnerable period.

Summarizing the discussion about o_rxtx (Figure A.7), its last operation is to invoke the *medium subnet* (BUSY) and put a token in place start_packet. This enables the timed activity t_Tpacket, which represents the transmission of a data frame. The duration of this activity depends both of the size of the transmitted data and the characteristics of the Physical layer. Details can be found in [7, 8]. When Tpacket completes, there are two probabilistic possible cases that depends on both the values of the places Interference/Collision and the data packet length (bits). The first case occurs when the place Interference is marked, which indicate the existence of external interferences during frame transmissions. Thus, the output gate o_packet_e is executed. This gate implements an algorithm that assumes that scenario 1 or 2 can occur. Conversely, the second case occur, and the output gate o_packet_OK is executed. This gate implements an algorithm that assumes that scenario 1 or 3 can occur. The algorithms implemented in these gates are represented in Figure A.10.

1. There was a collision during the data frame transmission: Collision is marked and the marking of Interference is indifferent. It was assumed that a collision always destroys the contents of the frames involved. Therefore, no frame is received by the remaining stations;

2. There were external interferences but no collisions during the frame transmission (only one station was transmitting): Collision is unmarked and Interference is marked. It was assumed that the data frame is corrupted and that frame errors are always detected by receiving stations. Besides, it was also assumed that the *PLCP Header* [8, 7] is always absent of errors. Although this is a slight optimistic assumption, it implies that receiving stations are always able to detect frame errors, which simplifies the modeling process. Otherwise the frame is not validated by the Physical layer at the receiving stations, which may lead to misinterpretations. In this case the transmitted frame is interpreted as a busy medium and not as a frame (even with errors). This problem and all consequences are discussed in [129]. Among the most important consequences is the incorrect implementation of this behavior in the NS-2 simulator;

3. The frame was transmitted successfully: Collision and Interference are both unmarked.

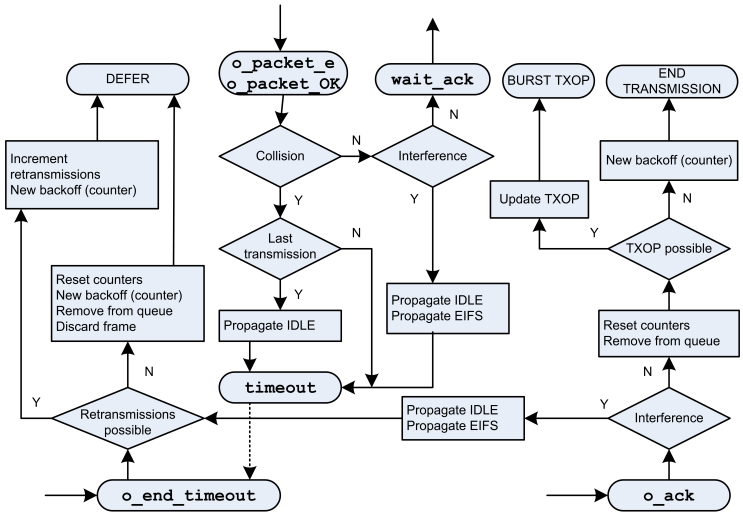

Figure A.10: output gate algorithms.

In scenarios 1 and 2, since there is no Acknowledge frame (data frame is corrupted or destroyed) the transmitting station should wait until the end

of a *timeout* interval. However there are differences of behavior between these scenarios. In scenario 1, the marking of place Ongoing is decremented by one in each model (replica) involved in the collision. When their marking reaches 0, it is guaranteed that the last transmission has finished. So, the *medium subnet* is invoked (IDLE) and the station goes to *timeout*.

In scenario 2, there are transmission errors which lead to different behaviors among stations. The transmitting station, as it cannot hear its own transmission (*i.e.* cannot detect errors) invokes the *medium subnet* (IDLE) and goes to *timeout*. However, receiving stations should detect the error and defer for EIFS (Extended Interframe Space). This implies that the remaining stations not involved in the transmission will also be affected. This aspect is generally misunderstood and is generally implemented incorrectly by the existent simulation models [129], including by the available NS-2 model.

The previous behavior is implemented by the *EIFS propagation subnet* (Figure A.11). In this case the transmitting station marks the commonplace Start_EIFS with N-1 tokens, where N is the number of replicas (stations). The input gate i_st_EIFS will enable the activity t_st_EIFS if the marking of Start_EIFS is >0, eifs isn't marked and the station is not transmitting. Therefore, all the replicas, except the transmitting one, will have this activity enabled.

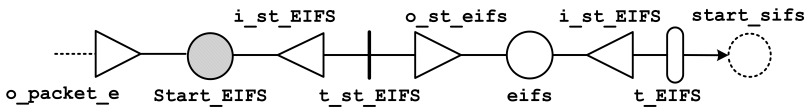

Figure A.11: EIFS propagation subnet.

When t_st_EIFS fires the output gate, o_st_eifs is executed, performing the following operations: the marking of Start_EIFS is decremented by one, place eifs is marked and local queue status goes to DEFER. Note that this activity will fire simultaneously in all replicas, after which the marking of Start_EIFS is 0.

The input gate i_st_EIFS guarantees that the timed activity t_EIFS is enabled only if eifs is marked and medium is IDLE, otherwise is disabled. Their duration corresponds to the EIFS interval ([8], pp. 85). When this activity completes eifs is unmarked and start_sifs is marked, beginning a new *slot counting* procedure. This subnet will therefore guarantee that each replica (station) involved defer for EIFS.

The *timeout* is modeled by the *timeout subnet* (Figure A.12). Place timeout is marked only in scenarios 1 and 2 by the execution of o_packet_e gate. The timeout interval is modeled by the timed activity t_Timeout. Their value is usually defined using an arbitrary value. However in [129] it is referred that this value is in fact defined by the standard as *aSIFTime* + *ACKLength*, where *ACKLength* is the duration of the transmission of an acknowledgement frame (this fact was confirmed by us (In [8], pp. 495). This problem and their consequences are discussed in [129], including the incorrect implementation of this behavior in the NS-2 simulator.

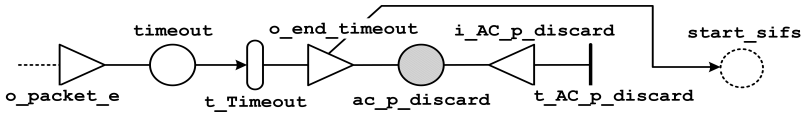

Figure A.12: Timeout propagation subnet.

When this activity completes the output gate o_end_timeout is executed. This gate implements an algorithm that is presented in Figure A.10. If the frame is discarded, place ac_p_discard(i) is marked. The subnet associated to the frame discarding has a behavior similar the queue discarding subnet already presented (this subnet is also invoked in o_AC_function algorithm). Moreover, the marking of qac, txop and txop_st_time are reset, and if medium is IDLE place start_sifs is marked, which begins a new *slot counting* procedure.

Summing up the previous discussion about o_packet_e or o_packet_ok execution, if scenario 3 is true then the transmitting station will execute the gate o_packet_ok, and it will be waiting by an Acknowledge frame sent by a responding station. In this case, place wait_ack is marked, which enables the timed activity t_Tack (Figure A.13). This activity represents the time necessary to transmit an acknowledgement frame and it has characteristics similar to t_Tpacket. When it completes, the output gates o_packet_e or o_packet_OK is executed. This gate implements an algorithm that is also presented in Figure A.10. In this case one of two scenarios can occur:

1. The acknowledgement frame was received without errors: Interference is unmarked. It is executed the o_packet_OK gate;

2. There were interferences during the transmission of the acknowledge frame: Interference is marked. It is executed the o_packet_e gate;

Scenario 2 leads to a situation where stations should defer for EIFS. However there is an important difference about EIFS behavior. In this scenario all the stations, except the one that sent the Acknowledge frame, defer for EIFS, which includes station represented by the current model. This means that the current model plus N-2 models defer for EIFS, while one replica does not (it goes to DEFER). This was implemented by marking eifs in the current model, and by using the EIFS subnet discussed previously to update the remaining models. However in this situation the place Start_EIFS is marked with N-2 tokens. The *medium subnet* is also invoked (IDLE).

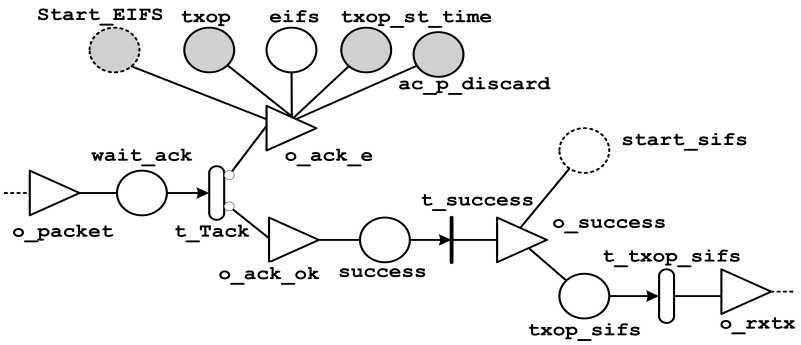

Figure A.13: Acknowledge subnet.

Scenario 1 corresponds to a situation where the frame exchange sequence was performed without any problems. Besides some housekeeping operations the output gate o_ack_ok verifies the conditions for a multiple frame transmission (TXOP). If there are packets in the queue and there is enough time in the TXOP timer to transmit another packet including its response frame, then the conditions are verified and the queue status is changed to BURST_TXOP. This is temporary identifier used only for distinguishing purposes. Otherwise this is the final transmission by the TXOP holder. In this case the queue status is changed to END_TRANSMISSION (also a temporary identifier) and the marking of qac, txop and txop_st_time are reset. In both situations place success is marked. This place is only a model artifact that enables to evaluate measures about successful frame exchange sequences.

When place success is marked the immediate activity t_success fires executing the output gate o_success. In this situation one of two scenarios can occur:

1. The queue status is BURST_TXOP. In this case the next data frame is transmitted immediately after *aSIFSTime*. In the model the queue

Table A.1: Example of performance measures.

Measure	**R(rate, place) I(impulse, activity)**
Throughput: frames, success	I(1, t_success) (1)
Queue: mean size	R(num. tokens, ac_q(i)) (2)
Queue: frames, discarded	I(1, t_AC_q_discard) (3)
Transmission: frames, discarded	I(1, t_AC_p_discard) (4)
Queue: waiting time	(2) / (Frame arrival rate – (3) – (4))

status is changed to TRANSMITTING, the TXOP timer is updated and place txop_sifs is marked. This enables timed activity t_txop_SIFS which has duration of *aSIFSTime*. When it completes the output gate o_rxtx is executed. This corresponds to the conditions in the beginning of a new data frame transmission (Figure A.7);

2. The queue status is END_TRANSMISSION. This is the last transmission. The *medium subnet* is invoked (IDLE), queue status is changed to DEFER and place start_sifs is marked, which begins a new *slot counting* procedure.

A.4 Performance Measures

Performance evaluation is performed by defining a set of measures in the model. In the context of SPNs these measures are derived from the concept of *reward* [138, 141]. Two types of rewards can be defined: *Rates*, associated with markings of the SPN, which are collected during the time the SPN resides on the marking. With this type of measure is possible to obtain occupation probabilities, average number of tokens at each place, etc. The other reward is *Impulses*, associated with transitions firings which are collected when the transition fires. With this type of measure is possible to obtain throughput values, frames discarded, etc (Table A.1). Several other measures can de derived from the previous ones. An example is the average waiting time in the queue, which can be obtained from Little's law [143].

A.5 Model Validation

The IEEE 802.11e Working Group (WG) was established in 1999. The IEEE 802.11 was published as an amendment to the original standard only

Table A.2: Parameters for verification.

	High	Medium	Low
AIFSN	2	4	7
CWmin	7	10	15
CWmax	7	31	255

on December 2005. During the standardization process a lot modifications were incorporated from early draft versions to the final one. Besides, it was developed a number of models during this period representing the behavior of EDCF (referred in draft version) or EDCA (as referred in the final version) function associated to QoS stations [86, 144, 93, 145, 146].

This section discusses the validation of the SPN EDCA model by showing its functionality with traffic of different ACs and its behavior when system changes from non- to the saturated area. The model was validated from two perspectives. Initially, and most importantly, the model was intensively debugged using several techniques. Firstly, it was introduced additional checks and outputs in the code, in order to point out the bugs. Secondly, pre-defined inputs were used to activate individual functions (parts of the model), which enable to check their internal behavior. Moreover, this data was used for a systematic comparison against the IEEE 802.11e specification [8].

Finally, it was compared our obtained results with results obtained by other authors through simulation. In this task, we followed the same methodology applied in [145, 146], where the authors validated both the EDCF and EDCA TKN models against the results presented by Mangold [144]. Mangold has implemented an EDCF simulation model and conducted some performance evaluations. It utilizes the IEEE 802.11a-PHY with a data rate of 24Mbps.

For the test of the SPN EDCA we chose the same simulation scenarios of [144, 145, 146], where the throughput of each traffic category (TC) is the metric compared in all simulations. The scenario consists of 3 different TCs: a high-priority isochronous flow with 128kbps and 80 bytes for data payload; a medium and a low priority Poisson flow with each 60kbps and 200 bytes for data payload. The backoff parameters are shown in Table A.2.

The number of stations is continuously increased from 1 to 15. All stations are in the range of each other and, it is assumed that there is no node mobility. Figures A.14, A.15 and A.16 show the results for the high, medium and low priority levels, respectively.

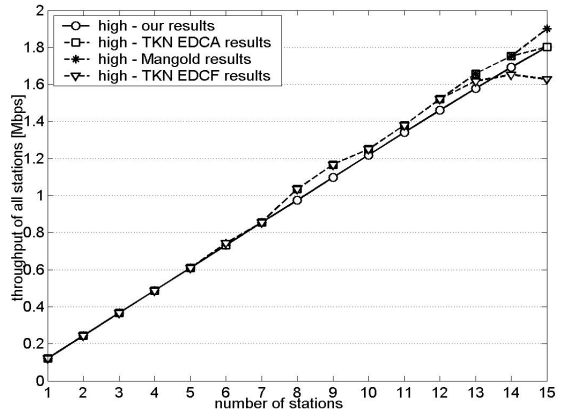

Figure A.14: Accumulated MAC layer throughput of all stations - high priority.

It is clear that the implemented SPN model (our results) achieves similar results to Mangold's work [144]. Furthermore, our results are also similar to TKN ones. It is worth noting that the TKN's simulation models are widely used by the scientific community. Therefore, it is assumed it as verified and "correct".

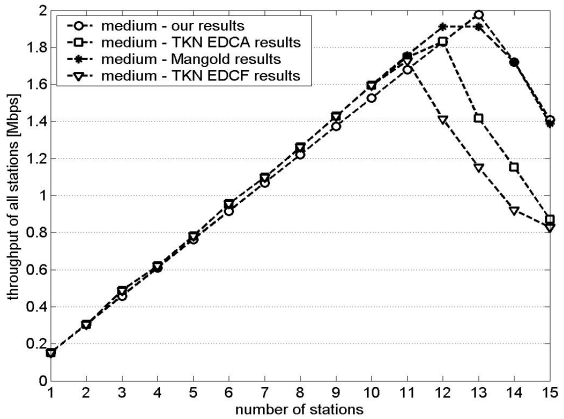

Figure A.15: Accumulated MAC layer throughput of all stations - medium priority.

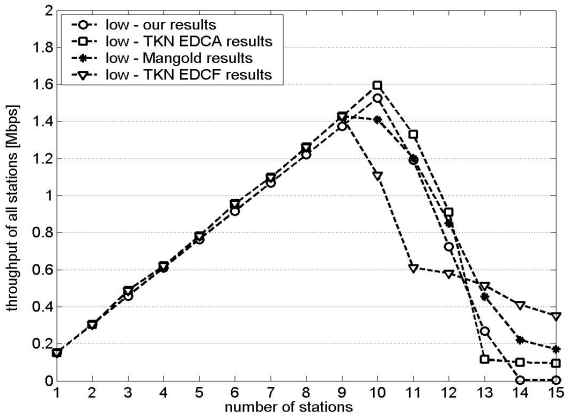

Figure A.16: Accumulated MAC layer throughput of all stations - low priority.

A.6 Summary

A Stochastic Petri Net simulation model that describes the dynamics of the Contention-Based Channel Access (EDCA) function of IEEE 802.11e was proposed. The model implements the main EDCA function in great detail, following closely the standard specifications. When compared with currently used simulation models, it provides a more accurate implementation of several aspects (*e.g.* timeouts and EIFS behaviors), a flexible implementation, and an easiness of use, which includes the possibility to obtain a different number of performance measures. It is also the first SPN model that covers this protocol.

Appendix B

List of publications

This Appendix presents the list of publications that are specifically used in this thesis (in reverse chronological order).

B.1 Journal Publications

1. **VTP-CSMA: A Virtual Token Passing Approach for Real-Time Communication in IEEE 802.11 Wireless Networks**, R. Moraes, F. Vasques, P. Portugal and J. A. Fonseca. To appear in IEEE Transactions on Industrial Informatics (ISSN 1551-3203), vol. 3, no. 3, August 2007. 10 pp.

2. **A Forcing Collision Resolution Approach Able to Prioritize Real-Time Traffic in CSMA-based Networks**, R. Moraes, F. Vasques and P. Portugal. Submitted to Computer Communications (ISSN 0140-3664).

3. **A Traffic Separation Mechanism (TSm) Allowing the Coexistence of CSMA and Real-Time Traffic in Wireless 802.11e Networks**, R. Moraes, F. Vasques, P. Portugal and J. A. Fonseca. In WSEAS Transactions on Communications (ISSN 1109-2742), vol. 5, no. 5, pp. 890-897, April 2006.

B.2 Conference Publications

1. **Real-Time Communication in 802.11 Networks: Timing Analysis and a Ring Management Scheme for the VTP-CSMA Architecture**, R. Moraes, P. Portugal, S. Vitturi, F. Vasques, and P. Souto. Accepted to be presented at 32nd IEEE Conference on Local Computer Networks (LCN), 2007.

2. **Real-Time Communication in 802.11 Networks: The Virtual Token Passing VTP-CSMA Approach**, R. Moraes, F. Vasques, P. Portugal and J. A. Fonseca. In Proceedings of the 31st IEEE Conference on Local Computer Networks (LCN), Tampa, Florida, USA, pp. 389-396, November 14-17, 2006.

3. **Simulation Analysis of the IEEE 802.11e EDCA Protocol for an Industrially-Relevant Real-Time Communication Scenario**, R. Moraes, P. Portugal and F. Vasques. In Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Prague, Czech Republic, pp. 202-209, September 20-22, 2006.

4. **A Stochastic Petri Net Model for the Simulation Analysis of the IEEE 802.11e EDCA Communication Protocol**, R. Moraes, P. Portugal and F. Vasques. In Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Prague, Czech Republic, pp. 38-45, September 20-22, 2006.

5. **An Enhanced Traffic Separation Mechanism to Support Real-Time Communication in IEEE 802.11e Networks**, R. Moraes, F. Vasques, P. Portugal and J. A. Fonseca. In Proceedings of the 24th Brazilian Symposium on Computer Networks (SBRC), Curitiba, Brazil, pp. 1075-1078, May 29 - June 2, 2006.

6. **A new Traffic Separation Mechanism (TSm) in Wireless 802.11e Networks: A simulation study**, R. Moraes, F. Vasques, P. Portugal and J. A. Fonseca. In Proceedings of the 6th WSEAS International Conference on Robotics, Control and Manufacturing Technology (ROCOM), Hangzhou, China, pp. 107-112, April 16-18, 2006.

7. **Probabilistic Timing Analysis of the h-BEB Collision Resolution Algorithm**, R. Moraes and F. Vasques. In Proceedings of

the 6th IFAC International Conference on Fieldbus Systems and their Applications (FET), Puebla, Mexico, pp. 107-114, November 14-15, 2005.

8. **Interference Caused by the Insertion of an h-BEB Station in Standard Shared-Ethernet Networks: Simulation analysis**, R. Moraes and F. Vasques. In Proceedings of the 2005 European Simulation and Modeling Conference (EUROSIS), Porto, Portugal, pp. 503-508, October 24-26, 2005.

9. **Real-Time Communication in Unconstrained Shared Ethernet Networks: The Virtual Token-Passing approach**, F. Carreiro, R. Moraes, J. A. Fonseca and F. Vasques. In Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Catania, Italy, vol. 1, pp. 425-431, September 19-22, 2005.

10. **Real-Time Traffic Separation in Shared Ethernet Networks: Simulation analysis of the h-BEB collision resolution algorithm**, R. Moraes and F. Vasques. In Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), Hong Kong, China, pp. 89-92, August 17-19, 2005.

11. **Arquiteturas de Qualidade de Serviço (QoS) para Suporte de Comunicação em Sistemas de Controle via Redes**, R. Moraes and F. Vasques. In Proceedings of the 2o. Seminário de Estudantes Brasileiros em Portugal (SIEBRAP), Porto, Portugal, pp. 97-98, July 17-18, 2005.

12. **High Priority Traffic Separation in Shared Ethernet Networks**, R. Moraes and F. Vasques. In Proceedings of the 4th International Workshop on Real- Time Networks - RTN´2005, Palma Mallorca, Spain, pp. 17-20, July 5, 2005.

13. **A Probabilistic Analysis of Traffic Separation in Shared Ethernet Systems Using the h-BEB Collision Resolution Algorithm**, R. Moraes and F. Vasques. In Proceedings of the 13th International Conference on Real-Time Systems (RTS), Paris - France, pp. 74-96, April 5-7, 2005.

14. **A Quality-of-Service (QoS) Based Approach for the Communication Support in Network-Based Control Systems: An on-going project**, R. Moraes and F. Vasques. In Proceedings of the 11th IFAC Symposium on Information Control Problems in Manufacturing (INCOM), Salvador, Brazil, 6 p. April 5-7, 2004.