Universidade do Porto

Faculdade de Engenharia

FEUP

Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-Based Networks

By

Mário Jorge de Andrade Ferreira Alves

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

February 2003

Doctoral Committee:

Prof. José Marques dos SANTOS	Chairman
Prof. Lucia Lo BELLO Prof. José Alberto FONSECA	External Examiners
Prof. Pedro Ferreira do SOUTO Prof. Manuel Pereira RICARDO	Internal Examiners
Prof. Francisco VASQUES	Supervisor
Prof. Eduardo TOVAR	Co-Supervisor

Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-Based Networks

Abstract

The communication infrastructure of current Distributed Computer-Controlled Systems (DCCS) is usually based on fieldbus networks, since they provide adequate levels of performance, dependability, timeliness, maintainability and cost. Nevertheless, cabling starts to be an obstacle for an increasing number of industrial automation applications, which impose or benefit from the use of mobile devices such as handheld computers or transportation equipment.

Within this context, there is a trend to extend fieldbus systems with wireless capabilities, leading to hybrid wired/wireless communication networks, which must support wireless/mobile communications while still fulfilling stringent DCCS requirements. Wireless communications must cope with real-time and dependability features at least similar to the ones encountered in traditional (wired) fieldbus networks. The support of inter-cell mobility turns this task even more difficult, since mobile nodes must handoff between radio cells in a transparent way.

The main research objectives of this thesis are the specification of a hybrid wired/wireless communication architecture based on a standard fieldbus protocol (PROFIBUS) and the proposal of the appropriate mechanisms and approaches to support and guarantee real-time communications with such an architecture. For this purpose, several design approaches for the architecture of the hybrid fieldbus network are analysed. Particular focus is given to how real-time communications can be guaranteed over such a network and to how mobility can be supported without affecting the real-time performance of the hybrid network.

The interconnection of heterogeneous physical media in a broadcast network leads to traffic congestion (increasing queuing delays) in the Intermediate Systems (ISs). Therefore, an innovative mechanism for eliminating traffic congestion in the ISs is proposed, which permits to reduce and bound system turnaround times through the insertion of additional idle time before a master End System (ES) issues request frames. This is implemented by setting appropriate values for the PROFIBUS Idle Time parameters, for every master ES.

Additionally, a methodology to compute the worst-case system turnaround time and duration of message transactions is also proposed. This permits to set the PROFIBUS Slot Time parameter, for all master ESs in the communication network and enables the evaluation of the worst-case response time of message transactions.

Finally, the impact of inter-cell mobility in the timing behaviour of the network is addressed and a timing analysis of the adopted mobility management mechanism is carried out. This analysis permits to compute values for the mobility-related network parameters. It should also be stressed that neither the system architecture nor the methodologies proposed in this thesis impose any changes to the PROFIBUS protocol.

Keywords: Real-time communications; fieldbus networks; wireless networks; interconnection of heterogeneous communication networks.

Comunicação de Tempo-Real utilizando Redes Híbridas Cabladas/Rádio baseadas em PROFIBUS

Resumo

As infra-estruturas de comunicação dos sistemas controlados por computador actuais são normalmente baseadas em redes de campo (*fieldbus*), dado disponibilizarem níveis adequados de performance, confiança no funcionamento, comportamento temporal, capacidade de manutenção e custo. No entanto, as soluções cabladas começam a ser um obstáculo para um número crescente de aplicações de automação industrial, que potencialmente poderão beneficiar ou mesmo exigir a utilização de dispositivos móveis.

Neste contexto, existe uma tendência para dotar as redes de campo de capacidades de comunicação rádio. Visam-se, por isso, arquitecturas híbridas que terão, contudo, de fornecer comunicações sem fio e móveis, satisfazendo os mesmos requisitos de confiança no funcionamento e de tempo-real. O suporte de mobilidade entre células rádio torna este objectivo difícil.

Os principais objectivos de investigação desta dissertação são o da especificação de uma arquitectura de comunicação cablada/rádio baseada numa rede de campo normalizada (PROFIBUS) e o do desenvolvimento dos mecanismos e abordagens apropriados ao suporte de garantias de tempo-real. Nesse sentido, são analisadas várias abordagens para a arquitectura da rede híbrida.

A utilização de dispositivos de interligação (*intermediate systems*) funcionando como repetidores provoca congestionamento de tráfego (atrasos crescentes nas filas) nos mesmos. Com o objectivo de resolver esse problema, é proposto um mecanismo inovador de inserção de tempos mortos (*idle time*) entre transacções, recorrendo para o efeito à utilização dos dois temporizadores *Idle Time* do PROFIBUS.

Adicionalmente, é também proposta uma metodologia para o cálculo do pior tempo de reacção do sistema (*system turnaround time*) e para o pior caso de duração das transações. Dessa forma é possível configurar de forma apropriada o parâmetro *Slot Time* (tempo máximo de reacção) do PROFIBUS. O cálculo do pior tempo de reacção das transações permite determinar o pior caso de tempo de resposta das transações.

Por fim, é feita uma análise temporal do comportamento do mecanismo de gestão de mobilidade adoptado, permitindo determinar os parâmetros adequados ao suporte desse mecanismo.

Desta forma, obtém-se um conjunto de metodologias que, no seu todo, permitem o suporte de aplicações de tempo-real baseadas em infra-estruturas de comunicação híbridas.

Palavras chave: Comunicações de tempo-real; redes de campo; comunicações rádio; interligação de redes de comunicação heterogéneas.

Communication Temps-Réel en Réseaux Hybrides Câblé/Radio Basé en PROFIBUS

Résumé

Les infrastructures de communication des systèmes distribués commandés par ordinateur (DCCS) sont habituellement basées sur des réseaux de terrain (*fieldbus*), car ils fournissent niveaux appropriés d'exécution, de fiabilité, de réponse, d'entretien et de coût. Néanmoins, les systèmes câblés commence à être un obstacle pour un nombre croissant d'applications d'automation industriel, qui imposent ou bénéficie de l'utilisation des dispositifs mobiles.

Dans ce contexte, il y a une tendance de additionner des capacité radio à les systèmes de *fieldbus*. S'envisage les systèmes hybrides câblé/radio, qui doit offre caractéristiques de temps réel et de fiabilité au moins similaires à des réseaux traditionnels (câblé) de *fieldbus*. Le support de la mobilité inter-cellule tourne cette objective difficile d'obtenir.

Les principaux objectifs de recherche de cette thèse sont les spécifications d'une architecture de communication hybride câblé/radio basée sur un protocole standard de *fieldbus* (PROFIBUS) et des mécanismes et abordages appropriés pour soutenir des communications en temps réel. À cette fin, plusieurs approches de conception pour l'architecture du réseau hybride de *fieldbus* sont analysées.

L'utilisation des systèmes intermédiaires (répéteurs) dans cette réseau conduit à la congestion du trafic. Par conséquent, on propose un mécanisme innovateur basée en l'insertion des temps mort avant de envier messages, par plaçant des valeurs appropriées dans les temporisateurs *Idle Time* de PROFIBUS.

En plus, on propose également une méthodologie pour calculer, dans le pire cas, les délais de réaction et la durée des transactions des messages. Ceci permet d'ajusté le paramètre *Slot Time* de PROFIBUS. Le calcule de le pire cas de la durée des transactions de message permet l'évaluation du pire temps de réponse des transactions de message.

De cette façon, est obtenu un ensemble d'outils cruciales pour la garantie des besoins temporels dans les applications distribuées temps critique, où la distribution est supporté par une réseaux hybride câblé/radio.

Mots-clés: Systèmes temps-réel; communications temps-réel; réseaux industriels ; interconnexion des réseaux de communication hétérogènes.

para a Sónia,
para os meus Pais
e em memória do Eng. Mesquita Guimarães

Acknowledgements

It is impossible to put down in words all the feelings inside me, in such a moment of a lifetime. In fact, the number of people involved in this work, either in professional or personal contexts, goes far beyond these few lines.

First, I would like to express my gratitude to Francisco Vasques and Eduardo Tovar, for their remarkable supervision work. I would also like to thank the members of the doctoral committee for the time and effort invested in the evaluation of this Thesis, namely to the external examiners José Alberto Fonseca and Lucia Lo Bello, and to Pedro Souto, for such unusual dedication.

Several institutions have been involved in the support of this work. The EU, through the IST programme, Ministério da Educação, through the PRODEP programme and also FCT, FLAD and DEMEGI-FEUP. Particularly to the IST programme, for giving financial support to the RFieldbus project. The participation in this project was of paramount importance for this work. I must particularly thank to Klaus, Gerhard and Roland (Siemens), Jörg and Lutz (IFAK), Peter (Softing), Christos and Stavros (ISI) and Fabrice (ST2E), for our fruitful technical discussions.

The support of the Polytechnic Institute of Porto (IPP) and its School of Engineering (ISEP) was invaluable. Specially Vitor Santos (Chief of the Executive Board of ISEP), for having believed in our group. Also to my colleagues in my teaching group, for accepting extra work, particularly to its co-ordinators along the last 3 years: José Mesquita Guimarães (no longer with us), Isabel Abreu, António Castro Vide and Custódio Dias.

Thanks to all the people in the IPP-HURRAY! research group. Eduardo, for his outstanding co-ordination work, his impressive ability to solve problems and, most importantly, for our deep friendship. Miguel, for having accepted the token for co-ordinating the implementation of the Manufacturing Automation Field Trial of the RFieldbus project. Filipe and Nuno, for their remarkable capacity and imagination for putting things to work. Luís Ferreira, for the fruitful discussions and Berta, for her good humor. Also to Veríssimo, Sandra Machado, Bertil, Ana, Luís Marques and Sandra Almeida, for their encouragement.

Finally, I am most grateful to my family and friends. I particularly want to stress the support and encouragement given by my wife, my mother, my father, my sister and my brother. They have been always by my side, especially in the most difficult moments. I deeply appreciate your sacrifice.

Porto, 3rd of February, 2003

Mário Jorge de Andrade Ferreira Alves

Table of Contents

Chapter 1

Overview

This thesis addresses the design of a hybrid wired/wireless communication architecture based on a standard fieldbus protocol (PROFIBUS) and the proposal of appropriate mechanisms for supporting real-time communications using that architecture. This chapter gives an overview of the research context and objectives, and also outlines the major contributions of this work.

1.1. Introduction

Nowadays, an increasing number of automation systems requires the use of mobile devices. Industrial automation applications such as automated warehousing, process control and discrete manufacturing commonly require automatic guided vehicles and hand-held equipment, for example. These scenarios impose or benefit from the use of wireless communications. Moreover, wireless communication systems provide significant cuts in cabling and maintenance costs, ease in the installation of equipment in hazardous areas and important add-ons in terms of flexibility and ability to evolve.

Considering the evolution of wireless local area network (WLAN) technologies, which are targeted to office applications, it would seem reasonable to use these standardised systems (e.g. IEEE 802.11b) in industrial automation purposes as well. Indeed, some industrial applications already benefit from these standardised WLAN's. This trend is mainly stimulated by the fact that commodity Ethernet technology is already commonly found at the factory level of many automation systems.

However, several constraints for the use of wireless technologies at the cell level and, more acutely at the field level still exist. Insufficient performance, low level of dependability and the non existence of appropriate wireless Medium Access Protocols (MAC) protocols to ensure real-time behaviour are examples. When mobility of wireless nodes between radio cells is a requirement, these problems become even more acute.

In this context, one must stress that high performance, high dependability and real-time behaviour are main features of state-of-the-art (wired) fieldbuses. Therefore, even if the addition of wireless capabilities to fieldbuses may introduce some important features to automation systems, an important concern to the system designer is that these wireless extensions do not disrupt the above mentioned characteristics of fieldbus networks.

This thesis addresses the design of a hybrid wired/wireless communication architecture based on a standard fieldbus protocol (PROFIBUS), and the proposal and discussion of the appropriate mechanisms and approaches for supporting real-time communications using that architecture.

1.2. Research Context

Typically, a fieldbus network consists of several End Systems physically connected through a wired bus. In the context of this thesis, the term *End System* (ES) will be used to denote not only a wired but also a wireless node or station. A wireless network is therefore composed of a set of wireless ESs that intercommunicate either directly or indirectly via wireless (e.g., radio) channels. If the wireless ESs are able to intercommunicate directly, the wireless network is usually called an *ad-hoc wireless network*. If messages need to be relayed by a central device (usually called access point or base station), the wireless network is usually called a *structured wireless network*.

In case interoperability with wired (legacy) ESs is also required, there is the need for specific devices to interconnect wired and wireless ESs. In the context of this thesis, these devices are denoted as *Intermediate Systems* (ISs). Of course, the central devices previously mentioned for the case of the structured wireless networks are examples of ISs. However, in this thesis, we are more concerned with the use of ISs to support hybrid wired/wireless communication networks (Figure 1.1).

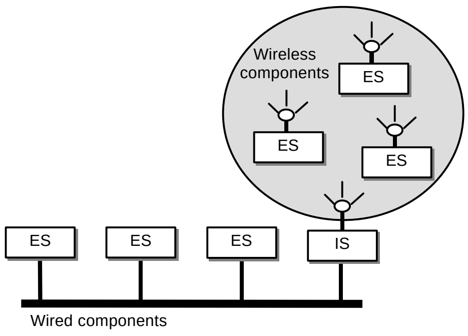


Figure 1.1: Example of a hybrid wired/wireless communication network

In many cases, the wireless ESs (nodes) will also be mobile nodes. It also happens that radio is the most suitable means to enable the interconnection between wireless components in industrial environments. Depending on the specific requirements of the application, the wireless/mobile nodes must be able to communicate while moving within a pre-defined three dimensional region. Issues such as the dimension and layout of the application, the existence of electromagnetic interference and the radio technology in use, may impose the division of this radio coverage region into a number of smaller regions, called radio cells.

Additionally, there may be situations where two or more wired fieldbus segments must be interconnected through wireless links. Therefore, the architecture of the hybrid wired/wireless network must support total interconnection between multiple wired segments and radio cells, and, also importantly, must provide appropriate mechanisms enabling inter-cell mobility of mobile ESs.

Such a hybrid architecture requires the analysis of a number of complex issues for which appropriate solutions must be devised.

1.3. Research Objectives

The main objectives of this thesis are *the specification of a hybrid wired/wireless communication architecture based on a standard fieldbus protocol and the proposal of the appropriate mechanisms and approaches to support and guarantee real-time communications with such an architecture*.

The hypothesis is that such an architecture is possible.

To this purpose, the following approach is going to be used. One important starting point is the selection of the wired fieldbus standard that is going to serve as the federating communication system in the architecture. Then, several approaches for hybrid architectures need to be analysed and discussed, namely concerning the type of Intermediate Systems to be used and the mechanisms to support inter-cell mobility.

Since most Distributed Computer-Controlled Systems (DCCS) for industrial applications impose stringent requirements, these architectural approaches must be assessed against features such as complexity, reliability and temporal behaviour. Particularly focus must be given to how real-time communications can be supported over such a hybrid network.

1.4. Research Contributions

The main contributions of this thesis are:

1. The specification of an architecture for hybrid wired/wireless fieldbus networks. Such specification involves the analysis and comparison of different design alternatives, using PROFIBUS as the federating communication technology, the definition of the functional and timing behaviour of the Intermediate Systems and the analysis of adequate mechanisms to support inter-cell mobility.
2. The definition of an innovative mechanism for eliminating increasing queuing delays (traffic congestion) in Intermediate Systems. This mechanism permits to reduce and bound system turnaround times through the insertion of extra idle time before issuing request frames. This is implemented by setting appropriate values for the PROFIBUS Idle Time parameters, in master stations.
3. A methodology to compute the worst-case system turnaround time and duration of message transactions (with several ISs between initiators and responders). This permits to set the PROFIBUS Slot Time parameter, for all master ESs in the communication network. Additionally, the duration of message transactions enables the evaluation of the worst-case response time of message transactions, in such type of hybrid wired/wireless networks.
4. A timing analysis of the adopted mobility management mechanism. This analysis permits to compute values for the mobility-related network parameters. The most relevant are the number of radio beacons that must be issued by the ISs such as the mobile ESs are able to perform handoff, and the inactivity time that must be respected during the mobility management period, before resuming normal system operation.

1.5. Structure of this Thesis

This thesis is structured as follows. Chapter 2 gives an overview of the state-of-the art in the area of communication networks for DCCS. Fieldbus networks are presented as the most prominent candidates to meet the particular set of requirements imposed by DCCS, although Industrial Ethernet is gaining increasing potential. Also, there is an increasing eagerness to use wireless communications in DCCS, leading to heterogeneous systems with full interoperability between wired and wireless nodes. Chapter 3 focuses on the federating communication technology for the proposed hybrid wired/wireless fieldbus network. The rationale for selecting the PROFIBUS protocol is presented by describing how it fulfils some important requirements. The most relevant characteristics of this protocol and a survey of the most important research work on PROFIBUS networks are also presented.

Chapter 4 addresses the architecture for the hybrid wired/wireless PROFIBUS-based network. First, the different components of such network are characterised and a set of rules for network configuration is defined. Then, several design alternatives are assessed and the repeater-based approach is elected as the preferred one for the proposed architecture. Finally, an innovative mechanism for supporting mobility that copes with simplicity and real-time requirements is introduced. Chapter 5 presents analytical models for the communication network, namely for the format of physical layer Protocol Data Units (PDUs) and for the timing behaviour of the Intermediate Systems (ISs). This model is the starting point for addressing the issues related to the timing behaviour of the Communication Network, which are then addressed in Chapters 6, 7 and 8.

Since the proposed architecture relies on different physical layers interconnected by repeaters, a suitable traffic adaptation mechanism is required in order to overcome traffic congestion problems (increasing queuing delays) in ISs. Chapter 6 presents an innovative solution to this problem that is based on an appropriate setting of the PROFIBUS Idle Time parameters. This idle time (inactivity period that a master must respect before issuing a request PDU or passing the token) guarantees that every message transaction has a small and bounded system turnaround time. Chapter 7 presents a methodology to evaluate the worst-case system turnaround time and the worst-case duration of message transactions. In Chapter 7, a methodology to set the PROFIBUS Slot Time parameter according to the requirements of the proposed architecture is also presented. Chapter 8 describes the impact of inter-cell mobility on the results presented in Chapters 7 and Chapter 6 and introduces a timing analysis for the mobility management mechanism (introduced in Chapter 4). This analysis enables the appropriate setting of crucial mobility parameters (related to the number of beacons transmitted by some ISs and worst-case duration of the mobility management period).

Chapter 9 shows how the approaches devised in chapters 5-8 can be put into practice. Two example scenarios, with and without inter-cell mobility, are elaborated. Then, some complementary simulation results are analysed, that enable drawing some additional conclusions about the proposed approaches. All the simulations were carried out using a system planning software application developed in the context of this thesis, which is based on the algorithms presented in Annexes B, D and E. The thesis concludes with Chapter 10, which summarises the presented contributions and identifies topics for future research. A list of acronyms and symbols is provided in Annex G.

Chapter 2

Survey of Technologies and Related Work

This chapter essentially provides an overview of communication networks for Distributed Computer-Controlled Systems (DCCS). Traditional fieldbus networks and, more recently, Industrial Ethernet networks are examples of technologies able to meet the particular set of requirements imposed by DCCS. Therefore, these technologies receive the most attention in this chapter. Additionally, wireless technologies are also addressed. In this thesis, we are particularly interested in analysing systems requiring interoperability between wired and wireless nodes (End Systems). This introduces the need to assess different options for the interconnecting devices (Intermediate Systems).

2.1. Introduction

This chapter addresses the current state and trends in communication networks for Distributed Computer-Controlled Systems (DCCS) in industrial environments.

Factory communication systems have suffered significant changes over the last 20 years or so. Local Area Networks (LANs) have substituted point-to-point communications, initially due to big savings in wiring and maintenance costs. The increasing decentralisation of measurement and control tasks, as well as the increasing use of intelligent microprocessor-controlled devices in industrial computer-controlled systems triggered the proliferation of fieldbus networks. A fieldbus network is a specific type of LAN aimed at the interconnection of sensors, actuators and controllers in applications ranging from manufacturing, process control, building automation and in-vehicle control.

More recently, there is an eagerness to use Ethernet-based technology in industrial environments. Factors contributing to this are its low price, maturity and stability. Moreover, Switched-Ethernet introduces functionalities with enormous potential for DCCS, such as congestion control, message prioritisation and micro-segmented full-duplex operation.

There is also an enormous eagerness to extend traditional fieldbuses with wireless and mobile communication capabilities. However, existent wireless communication networks have several constraints, namely low level of dependability and unpredictable timing behaviour. As state-of-the-art technologies such as IEEE 802.11 (WLAN - Wireless LAN) (IEEE 802.11, 1997) and 802.15 (WPAN - Wireless Personal Area Network) (Bluetooth Specification, 1999) become mature, their potential for DCCS applications grows, at least in what concerns their physical layer. A focus of interest will be the interoperability between these incoming wireless networks and already existent wired networks (e.g. fieldbus and Industrial Ethernet networks).

Reflecting this eagerness to extend current fieldbus networks with wireless capabilities, several companies are already supplying limited wireless extensions to major fieldbus networks (e.g. WorldFIP, PROFIBUS and CAN). Additionally, recent research is being conducted to develop solutions for wireless connectivity in industrial environments. For instance, (Alves *et al.*, 2000d), specifies a hybrid wired/wireless architecture based on the PROFIBUS protocol.

As a matter of fact, heterogeneity is not only a result of considering hybrid wired/wireless networks. Industrial communication networks are often heterogeneous, due to the coexistence of field-level and higher level networks, dissimilar field-level networks and separated domains of the same field-level network. In all these cases, interoperability is mandatory and must be achieved through the use of appropriate interconnecting devices acting as repeaters, bridges, routers or gateways. Throughout this thesis, devices that support end-user applications/services are referred to as *End Systems* (ESs), and devices that are used for network interconnection are referred to as *Intermediate Systems* (ISs).

The remainder of the chapter presents an overview of the communication network protocols suitable for implementing DCCS, of the most prominent wireless communication protocols for use in industrial environments and of solutions for interconnecting heterogeneous communication networks.

2.2. Communication networks for DCCS

2.2.1. General requirements

MAP/MMS (Manufacturing Automation Protocol stack with the Manufacturing Message Specification application layer) resulted from one of the most significant efforts in industrial communication networks developed in the early 80's. MAP/MMS implemented the full ISO/OSI layers model. This was eventually one of the reasons why applications based on MAP/MMS were costly, extremely complex to implement and integrate (Lederhofer *et al.*, 1996), a lot of computing power was required and the applications were restricted to Shop-Floor and Cell network levels. DCCS applications impose a number of specific requirements to the underlying communication networks (Decotignie, 2001), namely:

- ability to handle very short messages in an efficient manner (low overhead);
- ability to handle both periodic (sampling) and aperiodic (events) traffic with bounded message response times;
- ability to operate in harsh environments (EMI, vibrations, corrosion), providing an adequate level of dependability (mainly reliability and safety);
- low cost (except in safety-critical systems), including acquisition, installation, commissioning and maintenance expenses.

Soon was realised that MAP/MMS was not able to meet these requirements, contrarily to the so-called fieldbus networks. Similarly to other types of LANs, fieldbus networks are based on a layered structure derived from the seven-layer OSI model. However, due to the specialised requirements that must be met, the use of a full seven-layered architecture was precluded. Typical fieldbus network protocols implement a three-layer structure - physical layer, data link layer and application layer - even if some of these layers embody functionalities similar to those found in the other four layers of the OSI reference model.

Crucial parts of the fieldbuses (notably medium access control protocols) are designed to meet the temporal requirements of DCCS. The Medium Access Control (MAC) of Ethernet (CSMA/CD) was until recently one of the major drawbacks of Ethernet technologies. More recently, Switched-Ethernet appeared as a potential candidate for supporting DCCS, due to characteristics such as determinism.

2.2.2. Fieldbus standards – current state

The wide deployment of intelligent sensors, actuators and controllers triggered the proliferation of fieldbus systems. As a consequence of the difficulty to achieve a truly international fieldbus standard, in 1995 the CENELEC (European Committee for Electrotechnical Standardisation) proposed an interim European standard, comprising the three national standards existing in Europe: P-NET, PROFIBUS and WorldFIP. This initiative led, in 1996, to the EN 50170 standard (EN 50170, 1996). Although this European standard is a set of non-compatible profiles, it simplifies the choice in Europe, from several tens of fieldbus options down to three. In March 2000, a fourth profile was added to EN 50170: the Foundation Fieldbus. The Fieldbus Foundation was formed in 1994 from a merging of WorldFIP North America and the Interoperable Systems Project, having introduced its own specification in 1996.

CENELEC has also standardised Interbus (EN 50254, 1996) and DeviceNet on top of CAN (EN50325). CAN was standardised by ISO (ISO 11898, 1993) as a "Road Vehicle - Interchange of Digital Information" system and since then it is a standard for in-vehicle applications. Nevertheless, due to some interesting characteristics, CAN is also being considered for DCCS in industrial environments (Zuberi and Shin, 1997).

Having failed the definition of a unique fieldbus standard due to a number of reasons (Dietrich and Sauter, 2000; Pinto, 1999), in 2000 the IEC was able to ratify the Fieldbus Standard IEC61158 – Fieldbus Standard for use in Industrial Systems. It currently defines protocols and services for ControlNet (Type 2), PROFIBUS (Type 3), P-Net (Type 4), Foundation Fieldbus (Type 5), SwiftNet (Type 6), WorldFIP (Type 7) and Interbus-S (Type 8). Nowadays, it is commonly accepted that standard (and other) fieldbuses will continue to coexist in the future.

As the standardisation process has now stabilised, the emphasis is moving to the “User Layer”, namely addressing the definition of standard function blocks, descriptive languages, data quality metrics and interoperability (BSI, 2000).

2.2.3. “Industrial Ethernet” standards – current state

The first Ethernet specification was approved and released as IEEE Std 802.3 back in 1983 (IEEE 802.3, 1998). Since then, Ethernet has become the most popular, mature and low cost LAN technology. Recently, there have been some promising technological breakthroughs in Ethernet. Switched-Ethernet has been replacing Shared-Ethernet, due to a number of improvements. Switched-Ethernet separates traffic in smaller collision domains and, in micro-segmented full-duplex operation mode, collisions can be totally avoided (default mode in the proposed IEEE 802.3ae standard for 10 Gbit/s Ethernet).

Several international standards (e.g. IEEE 802.1p, 802.1D, 802.3x) provide Switched-Ethernet with powerful mechanisms to fulfil the dependability and timing requirements commonly found in DCCS. IEEE 802.1p (IEEE 802.1D, 1998) gives Layer 2 switches the ability to prioritise Ethernet traffic, while the IEEE Std 802.3x Flow Control mechanism provides means to control the generated traffic. All these mechanisms may be exploited to improve the timing properties of Ethernet networks. Switched-Ethernet can also be exploited to support dependable real-time systems. The IEEE 802.1D standard (1998), specifies the Spanning Tree Protocol, used to provide redundant network paths. Port Trunking establishes backbone links by treating multiple links as a single network pipe, also providing link redundancy.

Virtual LANs (IEEE 802.1Q, 1998) are also very promising to deal with flexibility and mobility. Using VLANs, a workgroup can be defined not by physical locations, but by multicast filters. If a device moves within the switched network, multicast filters automatically change, maintaining logical connectivity. Furthermore, other features such as dynamic switching, auto-negotiation, automatic load balancing can also be exploited to enable the dynamic behaviour of current and future DCCS.

Several international associations constituted by technology providers and end-users are pushing “Industrial-Ethernet” solutions forward. The Industrial Automation Networking Alliance (IAONA) and its European-based sister organisation (IAONA Europe), aim at establishing Ethernet as the standard in the industrial environment. The Industrial Ethernet Association (IEA), formed in 1999, intends to establish standards for the use of Ethernet products in the industrial marketplace. The 10 Gigabit Ethernet Alliance (10GEA) is working towards the 10 Gbit/s Ethernet standard that will provide a collision-free behaviour in a fibre-optic transmission medium. The Fieldbus Foundation has defined the High-Speed Ethernet (HSE), as the backbone for interconnecting traditional (low-speed) Foundation Fieldbus segments.

A lot of effort is being dedicated to the consolidation of Ethernet as a communication network for DCCS, both in the academic (Rüping *et al.*, 1999; Alves *et al.*, 2000b; Kweon and Shin, 2000; Jasperneite and Neumann, 2001; Varadarajan, 2001; Lo Bello and Mirabella, 2001a) and technology providers (e.g. Hirchmann, Rockwell, AEG/Schneider, Siemens) communities. Nevertheless, as the idea that “Industrial Ethernet” will replace current higher level fieldbuses (e.g. PROFIBUS-FMS) gets wider acceptance, it is foreseen that it will not substitute lower level fieldbuses (Dietrich and Sauter, 2000; Leblanc, 2000; Huselbos, 2001; Decotignie, 2001; Schiffer, 2001).

2.2.4. Wireless communication protocols for DCCS

There is an increasing eagerness to support wireless communications in DCCS for industrial environments. However, a strong requirement is that interoperability between wired and wireless nodes should be possible. Moreover, it is expected that wireless communications offer characteristics (e.g., performance, dependability) similar to current (wired) fieldbus networks and that these wireless technologies fulfil end-user expectations (e.g., radio coverage, speed of mobile devices).

(Alves *et al.*, 2000c) describes a set of basic requirements (bit rate, bit error rate (BER), range, path loss and delay spread), derived from a survey to several industrial users. The basic radio requirements express the inherent ability of the system to communicate efficiently via radio channels in (harsh) industrial environments. (Miaoudakis *et al.*, 2000) present the results of a thorough measurement campaign that was carried out in several manufacturing and process control industries. Their experimental work allowed setting adequate thresholds for these parameters (bit rates in the order of 2 Mbit/s, BERs below 10^{-5}, ranges greater than 70 m, path losses around 100 dB and delay spreads above 200 ηs).

Several wireless communication standards emerged in the last decade. IEEE has standardised two wireless communication protocols, one for Local Area Networks (WLAN) and the other for Personal Area Networks (WPAN). IEEE 802.11 (1997) recently split into two sub-standards – IEEE 802.11b (1999) and IEEE 802.11a (1999), providing maximum Physical Layer (PhL) bit rates of 11 Mbit/s and 54 Mbit/s, respectively. While products based on the IEEE 802.11a are only expected end 2002, a significant amount of commercial products based on IEEE 802.11b already exist. Bluetooth (1999) is a wireless communication protocol for short-range radio networks and has a bit rate of 1 Mbit/s (a 2 Mbit/s bit rate version is being specified). It was recently ratified in the IEEE 802.15.1 international standard ("Wireless MAC and PHY Specifications for Wireless Personal Area Networks (WPANs)".

The European Telecommunications Standards Institute (ETSI) has also been responsible for the standardisation of a set of wireless communication protocols. HIPERLAN (High PErformance Radio Local Area Network) versions 1 (EN 300652, 1998) and 2 (ETSI, 2000b) provide maximum bit rates of 20 Mbit/s and 54 Mbit/s, respectively. HIPERLAN1 technology has a negligible market share, when compared to 802.11a and 802.11b products. HIPERLAN2 seems to have a more promising future, since it is being harmonised with the IEEE 802.11 standards and it concentrates the interest for product development from the vast majority of companies.

The (EN 300175, 2001) standardises DECT (Digital Enhanced Cordless Telecommunications), which aims the access to existing wireless LANs and will soon reach a 2 Mbit/s maximum data rate. UMTS (Universal Mobile Telecommunication System) is intended for use in the third generation mobile communications (ETSI, 2000a), and offers a maximum data rate of 2Mbit/sec (under stationary conditions). Since it operates in a licensed band, charges have to be paid to an operator.

All the previously mentioned wireless communication protocols have strong technical features and exhibit good potential to satisfy most of the general requirements for a DCCS. However, they present several different features such as bit rates (e.g., bit rates ranging from 1 to 54 Mbit/s), target applications and availability of products. (Miaoudakis *et al.*, 2000) defend that only three of these wireless communication protocols satisfy all basic radio requirements for industrial environments: HIPERLAN2, UMTS and IEEE 802.11b. The reasons are briefly outlined next.

Considering the requirements imposed by DCCS in industrial automation, Bluetooth has an insufficient range (10 m) and, together with DECT, has insufficient bit rate. Although DECT has a provision to extend the bit rate to 2 Mbit/s, no products have been announced to support such bit rate yet. More advanced technologies like UMTS, HIPERLAN1 and HIPERLAN2 exhibit enhanced technical characteristics but there are hardly any products available today. Moreover, HIPERLAN1 it is vulnerable to large delay spreads and UMTS requires licensing costs (due to the frequency band) and regulation prohibits its usage unless the whole protocol stack is used, in accordance with the UMTS standard. Technologies based on the IEEE 802.11b standard fulfil the technical requirements and are mature. Nowadays, wireless LANs based on the IEEE 802.11b protocol have widespread use, and there is an increasing number of suppliers, commercial products available and installations.

2.2.5 Research on the use of wireless communication protocols in DCCS

There is a growing number of research works addressing the problem of using Commercial-Off-The-Shelf (COTS) wireless communication protocols in DCCS, mostly addressing the dependability (e.g., reliability and security), the real-time performance of the MAC layers and the interoperability between these and traditional (wired) networks.

(Cavalieri and Panno, 1997) proposed to interconnect IEC/ISA fieldbus networks via an IEEE 802.11 wireless LAN backbone (using bridges), introducing modifications to the 802.11 MAC in order to fulfil real-time requirements. (Pradhan and Chiueh, 1998) describe a complete architecture for the interconnection of Ethernet and WaveLAN (IEEE 802.11), including a mobility management mechanism. While in the wired part of the network the traditional distributed-token passing REther MAC is used, a new centralised token passing scheme (managed by each Base Station) is proposed for the wireless MAC. Bounded message delivery is guaranteed even when mobile ESs are performing handoff.

(El-Hoiydi and Dallemagne, 2000) analyse how inter-cell mobility impacts real-time performance in hybrid IEEE 802.3/802.11 networks. The same researchers are currently working on a dedicated layer to provide bounded message delivery latencies using the above standard MACs. A solution to this problem was presented in (Mock *et al.*, 2000), although interconnection with a wired network is not addressed.

(Koulamas *et al.*, 2001a) analyse candidate COTS wireless communication technologies against DCCS requirements and conclude that UMTS, IEEE 802.11b and HIPERLAN2 are the most suitable. They study the performance of a PROFIBUS message cycle for the three previously mentioned wireless communication technologies under different integration approaches. A similar work was carried out by (El-Hoiydi and Decotignie, 2001), but addressing the Bluetooth protocol. They present the delay performance of two-way (send data with acknowledge) transactions, under packet loss probability caused by interfering piconets.

In (Bilstrup and Wiberg, 2000), the authors analyse the performance of Bluetooth in industrial environments, showing its appropriateness for short-range wireless communication. The same authors broaden their work in (Wiberg and Bilstrup, 2001), providing an overview of wireless communications technology. They analyse the adequateness of IEEE 802.11, HIPERLAN2 and Bluetooth to fulfil the communication requirements of a diversity of industrial applications. They defend that as wireless protocols provide increasingly higher throughput, new and more complex coding schemes can be used (an innovative coding scheme – Deadline Dependent Coding – was proposed to overcome the problems of low reliability of wireless media).

2.3. Interconnecting heterogeneous communication networks

2.3.1. Interconnecting field level and higher level networks

During the “golden age” of MAP/MMS, several European projects (e.g ESPRIT Projects 5602 (FICIM), 5104 (CNMA) and 7096 (CCE-CNMA)) addressed the problem of interconnecting different industrial communication networks. A number of gateways to different networking levels were designed and developed. Examples include MAP/MMS-PROFIBUS/FMS (Marcos *et al.*, 1997), MMS-FILBUS (Tovar and Cardoso, 1995), MMS-GPIB (CNMA, 1991) and MMS-TCP/IP (CCE-CNMA, 1994). While these solutions might not have had widespread use in industry (due to the decline of MAP/MMS), the philosophy inherent to their architectures is very much similar to the one of current Internet/fieldbus gateways.

Nowadays, the main motivation for interconnecting field-level networks and higher level networks is the trend towards Internet access to the factory floor. The “I can access anything from anywhere” concept is definitely driving new strategies to tackle the communication requirements of the modern factory (Rockwell, 2000; Renner, 1999). Most fieldbus manufacturers (e.g. Hilscher, Deutschmann Automation, AEG/Schneider, HMS and Bihl&Wiedemann) provide Ethernet TCP/IP gateways, and many researchers are proposing solutions for Internet monitoring and maintenance of fieldbus networks (Wollschlaeger, 1997; Knizak *et al.*, 1997; Neumann and Iwanitz, 1997; Hutter and Steiner, 1999; Palenski and Sauter, 2000; Pratl *et al.*, 2001). With this interoperability, COTS user-friendly interfaces (e.g. browsers) may be used for monitoring/control (Hutter and Steiner, 1999) and management (Wollschlaeger, 1997; Knizak *et al.*, 1997) of fieldbus systems. Nevertheless, security issues must carefully analysed (Palenski and Sauter, 2000).

A common approach for an Internet/fieldbus gateway is to make data in the fieldbus accessible using protocols commonly used in the Internet, such as HTTP, FTP, SNMP, ICMP, CORBA, RMI, DCOM, OPC (Soucek *et al.*, 2000; Knizak *et al.*, 1997; Neumman and Iwanitz, 1997; Pratl *et al.*, 2001). (Hutter and Steiner, 1999) suggest that an Internet/P-NET gateway can be based on the VIGO platform, since VIGO controls P-NET and can be controlled via OLE (over TCP/IP). (Tovar *et al.*, 2001; Pacheco *et al.*, 2001; Ferreira *et al.*, 2001) propose innovative admission control and scheduling mechanisms that allow multimedia TCP/IP traffic to coexist with real-time control traffic in a PROFIBUS-based DLL protocol.

Several companies provide Ethernet/fieldbus gateways, permitting the access to process data over the Internet. Hilscher, Deutschmann Automation, AEG/Schneider, HMS and Bihl&Wiedemann provide Internet (TCP/IP on top of Ethernet) gateways to ControlNet, DeviceNet, PROFIBUS (DP, FMS), Interbus, CANopen, AS-I, ModBus, ModBus Plus and RS232/485 (Modbus RTU/ASCII). Most of the times, the gateway behaves as a master station in the fieldbus network, maintaining an updated image of process data to be accessed from the Ethernet network (proxy-like behaviour). Since at this point, there is no standard for the access to process data at the application layer (on top of TCP/IP), the ModBus protocol is commonly used as the application layer.

Due to the usage of the Modbus protocol on both sides of the intermediate system, AEG/Schneider proposes a “Modbus to Ethernet Bridge” and a “Modbus Plus to Ethernet Bridge”. Actually, these “bridges” maintain two internal tables mapping Modbus MAC addresses to IP addresses. While Ethernet end-systems using TCP/IP can function as Modbus masters, Ethernet may also be used just to connect multiple Modbus networks. Similarly, (Kunert, 1997) proposes to break up distance limitations in current fieldbus systems by interconnecting different PROFIBUS networks using an ATM backbone.

2.3.2. Interconnecting dissimilar field level networks

Although fieldbus systems are in widespread use in industry, there is still a significant number of devices that only communicate via a serial data interface (e.g., RS232), usually using Modbus/Modnet higher layer protocols. Several companies provide serial/fieldbus gateways (e.g. Hilscher’s PKV, Deutschmann Automation’s UNIGATE, HMS’s AnyBus) to integrate these legacy systems into several fieldbus networks. These gateways can operate in two different ways: either they maintain an internal image of the (serial) device to which they are connected, or each individual frame is converted directly between the two protocols.

Bihl&Wiedemann supplies a multitude of AS-I gateways that contain an AS-I master responsible for maintaining an image of AS-I slaves. A fieldbus master accesses this AS-I network image by communicating with the fieldbus slave (also) contained in the gateway. Similarly, Deutschmann Automation’s CANopen-PROFIBUS-DP gateway operates as a slave in the PROFIBUS network and as a master in the CANopen network. Claiming more than 2000 gateway combinations, X-Link gateways (from SST) apparently (available information is very restricted) convert frames between the two networks. Additionally, Beeston (2001) refers a pioneer WorldFIP-HART gateway implemented by Electricité de France R&D division (“Everest” tool for test and validation of smart sensors and actuators).

There is a very limited number of relevant scientific papers addressing this topic. (Sveda and Zezulka, 1997) have proposed several gateway architectures to interconnect different fieldbus networks. Within the context of the CAROSSE project, a CAN-VAN bridge was developed for the PSA Group and its behaviour was carefully analysed (Castelpietra *et al.*, 2000). While their work addresses in-vehicle applications, the underlying ideas may also be useful for using CAN in industrial automation systems.

2.3.3. Interconnecting different domains of the same field level network

A fieldbus system can be divided into a number of logically separated domains, i.e. nodes are grouped into a set of domains interconnected by devices with routing (address filtering) functionality (e.g., bridges). By an appropriate assignment of nodes to domains, traffic load can be significantly reduced in each individual domain, leading to a better network performance and to shorter message response times. Communication between two end-systems in one domain does not restrict communication between two end-systems in another domain. Error-containment is also improved, since a problem in one domain does not affect the others (e.g., lost token in token-passing fieldbuses).

P-NET and ModBus are examples of fieldbuses that were conceived with the previously described capability. P-NET Multi-Net Structure is based on "Multi-Port Masters", connecting logically separated domains. Moreover, these ISs behave like routers, since the P-NET protocol incorporates variable address length capabilities. This means that the address path defined between a master ES and a slave ES can include routing through ISs (Multi-Port Masters). Modbus/Modbus Plus Bridges (AEG/Schneider, 1996) have a functionality very similar to P-NET bridges. Each frame contains a routing path, in order for the ISs to know if they should parse the frame or not. Up to four ISs can be present in the message path between source and destination ESs. If a data message for a remote ES is received at one of the IS ports, the IS stores the message and then forwards it to a destination address (IS or ES) in the next network, as soon as it has received the token to transmit to that network. The IS sends an immediate acknowledgement to the originator of the message.

The PROFIBUS Standard defines an "extended addressing" scheme, but does not specify some fundamental aspects about traffic between nodes in different domains, namely the data transfer mechanisms and time-related issues. (Monforte *et al.*, 2000b) analyses the PROFIBUS Standard's guidelines for segmentation and proposes a "bridge-like" behaviour for the IS (somehow similar to the one proposed in (Kunert, 1997). In order to fulfil the industrial need to connect PROFIBUS-DP to PROFIBUS-PA networks, Siemens supplies the DP/PA Bus Coupler and DP/PA Link products. However, they opted for a "proxy-like" gateway approach in the PROFIBUS DP/PA Link and a "repeater-like" approach in the DP/PA Bus Coupler (Siemens, 2000). In the former, two different logical rings exist (one on DP and the other on PA). The PA master in the IS is responsible for maintaining an updated PA process data image which, in turn, may be accessed through the DP slave of the IS. On the other hand, the DP/PA Bus Coupler only translates 45.45 kbit/s asynchronous to 31.25 kbit/s synchronous messages (acting as a repeater, therefore with a unique logical ring).

2.3.4. Interconnecting wired and wireless networks

Finally, there is also an enormous trend to provide DCCS with wireless communication capabilities. Some recent achievements concerning research initiatives and currently available products in the area of wireless communications for DCCS are described next.

Research Initiatives

Within the RFieldbus project (IST-1999-11316, High Performance Wireless Fieldbus in Industrial Multimedia-Related Environment), a hybrid wireless/wired fieldbus system architecture has been devised (Alves *et al.*, 2000d). This approach is currently under implementation and will be validated and demonstrated in two field trials. In the Rfieldbus architecture, Intermediate Systems behave as repeaters, in order to maintain full compatibility with legacy PROFIBUS nodes and to minimise communication latencies and the complexity of the architecture (Alves *et al.*, 2002). Moreover, ad-hoc and structured wireless networks are possible, and a very simple but efficient mobility management mechanism allows inter-cell mobility (Alves *et al.*, 2002).

Two additional european projects addressed the extension of traditional fieldbus networks to support wireless communications – MOFDI and OLCHFA. MOFDI project

(ESPRIT 27035, Mobile Fieldbus Devices in Industry) objectives included providing an interconnection between wired and wireless ESs for several fieldbus systems, supporting real-time communications (Lawton, 2001). Unfortunately, although the project ended mid-2001, no significant information concerning project results is available. Even less information is available about the OLCHFA project (ESPRIT 7210, An Open Low-Cost Time-Critical Wireless Fieldbus Architecture), which ended in 1994. A hybrid wired/wireless fieldbus network based on FIP was envisaged, also with concerns as regards the support of time-critical communications.

A different approach was adopted in the W_2F project (Wireline/Wireless Factory/Facility Fieldbus) framework (supported by the Austrian START programme), since a completely new wired/wireless industrial communication protocol is being conceived from scratch (Schmid, 1999). This is a basic research project that does not rely on any already existent fieldbus systems, but rather on the use of a spread-spectrum CDMA technology on both wired and wireless communications. The use of a common MAC for wired and wireless communications allows to overcome the interoperability problems arising from the use of traditional fieldbus and wireless networks.

In (Wiberg and Svenson, 2000), the guidelines for a research project are proposed. Like the previously mentioned project, CDMA technology will be analysed in order to achieve a hybrid communication network for time and safety-critical applications.

Researchers from KVASER and Daimler-Chrisler carried out relevant work on the wireless extension of CAN networks using Bluetooth technology (Fredriksson, 1999; Wunderlich *et al.*, 2000). While their framework addresses automotive applications, they propose a MAC-less Bluetooth that may be applied to industrial real-time control systems. To this purpose, a customised hard real-time MAC (DLL) should substitute the default non real-time Bluetooth MAC. The IS would behave just like a bridge, converting between the producer-consumer model of CAN and the source-destination model of Bluetooth, if error detection/correction mechanisms would not be needed in the Application Layer. Due to this reason, the IS behaves as a gateway. To our best knowledge, no mechanism for inter-cell mobility is proposed within this framework.

(Morel *et al.*, 1995) proposes a proxy-like gateway architecture, where the produced and consumed variables from the wireless ESs are local variables in the IS. Although the authors propose a TDMA MAC for the wireless protocol, no wireless (physical layer) technology is suggested. Since multiple-cells are not considered, inter-cell mobility is not required in their approach.

(Willig, 1997) proposes an architecture able to support mobility between cells (although no mobility management mechanism is described), where different base stations (ISs acting as gateways) communicate via an ATM backbone. Nevertheless, the ATM network just serves to interconnect base stations, since only wireless stations are supported. A Flexible TDMA MAC is proposed for the wireless communication protocol. Recent work by the same author now addresses the interconnection between PROFIBUS and a wireless polling-based protocol, focusing on the real-time behaviour. The author argues that the PROFIBUS DLL is not adequate for wireless communications, since the probability of lost or corrupted tokens is non-negligible. This assumption is based on theoretical (Willig and Wolitz, 2001) and experimental work (Willig, 1999) on the behaviour of PROFIBUS over error-prone links, namely over a DSSS physical layer (similar to the one of IEEE 802.11).

(Lee and Lee, 2001; Lee *et al.*, 2002) propose a scheme for integration of IEEE 802.11 wireless nodes in a (mono-master) PROFIBUS-DP network. The approach is to use an application layer gateway acting as a protocol converter. The gateway performs a role of translator between the two protocols by converting the format of a data frame. In addition, the gateway implements a virtual polling algorithm at the Application Layer of the wireless protocol, to reduce the uncertainty involved in accessing the wireless network. The described experimental results show the feasibility of the proposed solution for industrial applications involving mobile devices.

Similarly to AEG/Schneider's "Modbus to Ethernet Bridge" and the proposal encountered in (Morel *et al.*, 1995), (Cavalieri and Panno, 1997) proposed to interconnect IEC/ISA fieldbus networks via an IEEE 802.11 wireless LAN backbone (using bridges). Nevertheless, some modifications to the 802.11 MAC are needed in order to fulfil real-time requirements. Previously, (Cavalieri *et al.*, 1994) proposed a similar solution using a FDDI backbone instead.

There are several proposals for the interconnection between (wired) Ethernet ESs and wireless ESs. Nevertheless, only a few (El-Hoiydi and Dallemagne, 2000; Pradhan and Chiueh, 1998) look for real-time guarantees. The former analyses how inter-cell mobility impacts real-time performance in hybrid IEEE 802.3/802.11 networks. Moreover, the authors are currently working on a dedicated layer to provide bounded message delivery latencies using the above standard MACs. A solution for this problem was presented in (Mock *et al.*, 2000), although interconnection with a wired network is not addressed. (Pradhan and Chiueh, 1998) describe a complete architecture for the interconnection between Ethernet and WaveLAN, including a mobility management mechanism. While in the wired part of the network the traditional distributed-token passing REther MAC is used, a new centralised token passing scheme (managed by each Base Station) is proposed for the wireless MAC. Bounded message delivery is guaranteed even when mobile ESs are performing handoff.

In (Decotignie *et al.*, 2001), the authors present an overview of different architectures for hybrid wired/wireless fieldbus systems.

Currently available products

Quite often, the available technical information about commercial products is very superficial. However, it is worthwhile to present some of their characteristics. ALSTOM provides a radio extension to WorldFIP networks (REKA120-WorldFIP, developed by ST2E) that has a very simple and transparent repeater behaviour. It permits to set up ad-hoc networks (just wireless End Systems), wired/wireless interconnection and redundancy over the wireless medium. Mobility management is not supported; the IS just converts between 1 Mbit/s synchronous transmission (wired) and 1.2 Mbit/s asynchronous transmission (wireless). Several companies supply infrared extensions to fieldbus networks (e.g. Hirchmann), which are only adequate for short length line-of-sight connections.

The Finish company Elektrobit has developed the WUCS – Wireless Underground Communication System, for the control of moving production equipment in underground mines. This system is composed of a wireless part based on a cellular network principle and of a fixed core wired network based on ATM technology, connecting Base Stations (interconnecting the wired and wireless networks) and end-systems dedicated to control.

WAVEcan (KVASER) provides a wireless extension to CAN, using Bluetooth technology. By default, the IS behaves as a transparent repeater. Nevertheless, it is possible to put the IS working as a bridge, filtering the message stream so that only a subset of the CAN messages are relayed to the air. A 125 kbit/s bit rate is supported. Importantly, KVASER refers that there is an internal delay in the IS, to be considered in a real-time system, but does not provide any detailed information on that subject.

2.4. Summary

This chapter presented an overview of communication networks for Distributed Computer-Controlled Systems in Industrial Environments. First, a brief history of fieldbus networks and standards and how Industrial Ethernet has recently appeared as a prominent candidate for supporting factory communications above field level was presented. Then, some standard wireless communication protocols that might pave the way for the integration of radio communications with legacy communication systems, while coping with dependability and real-time requirements were described. The issue of heterogeneity in factory communication systems has also been addressed and it has been surveyed how the interconnection of different communication networks can be achieved in the different approaches.

Chapter 3

Federating Communication System

This chapter focuses on the federating communication infrastructure for the hybrid wired/wireless fieldbus network. It starts by discussing the rationale beyond the appropriateness of the PROFIBUS protocol to play that role and by describing how this fieldbus protocol fulfils some important technical requirements. The characteristics of the PROFIBUS protocol that are most relevant within the scope of this thesis are then presented. The chapter ends with a survey on some of the most important research work on PROFIBUS networks.

3.1. Introduction

This thesis addresses the extension of a traditional (wired) fieldbus network to support wireless and mobile nodes. This hybrid wired/wireless network could potentially be based on different fieldbus systems, taking into account that their characteristics often overlap. PROFIBUS has however a number of advantages, since it gathers a set of features that are relevant for the targeted architecture. These features are described next.

PROFIBUS is the world's leading fieldbus standard for manufacturing automation and process control (over 20% market share). Since it is standardised under the Fieldbus Standards EN 50170 and IEC 61158, stability and openness for users and vendors are guaranteed.

PROFIBUS-DP, one of the PROFIBUS communication profiles, offers one of the fastest transmission speed available today in a fieldbus system (12 Mbit/s). This is an important characteristic to support multimedia bandwidth-consuming applications, which are more than ever a requirement in industrial DCCS. Moreover, and for that purpose, its ability to support significant amounts of data in the Data Link Layer (DLL) frame (246 bytes per frame) minimises the need for packet segmentation, if IP tunnelling is envisaged.

PROFIBUS has another interesting feature. It is designed to support high-priority and low-priority messages. This is most important when both time-critical and non time-critical traffic coexist. The PROFIBUS MAC protocol, being based on the measurement of the token rotation time, induces a well-defined timing behaviour for the transferred messages, since the token cycle duration is upper-bounded. Therefore, the PROFIBUS protocol is able to support real-time traffic, with bounded response times (Tovar and Vasques, 1999a).

The PROFIBUS ring maintenance mechanisms also seem appropriate to handle wireless/mobile stations. However, there are some aspects that must be carefully assessed. The PROFIBUS MAC is based on token passing between master nodes.

Therefore, the fact that some of the masters are wireless and mobile may cause some problems. Degraded radio quality and handoff (between radio cells) may endanger the token passing procedure, affecting the dependability and responsiveness of the network. This means that the wireless Physical Layer (PhL) must have an acceptable level of reliability (equivalent to the wired PhL), and the architecture must encompass a (inter-cell) mobility management mechanism that does not degrade dependability and responsiveness.

The error detection and correction mechanisms of the PROFIBUS DLL are potentially suitable to ensure an acceptable level of reliability to the application layer. The parity bit for every transmitted character, the start and end frame delimiters with Hamming Distance 4, the Frame Check Sequence field in the DLL frame and the Message Cycle Control are examples of such mechanisms.

It should be noted that the particular characteristics of the chosen fieldbus protocol (PROFIBUS, in our approach) impact the architectural decisions that are made in Chapter 4. The remainder of this chapter presents the most relevant characteristics of the PROFIBUS DLL protocol in the context of this thesis and surveys the most relevant research work on PROFIBUS, mainly concerning its real-time performance.

Throughout this thesis, parameters denoted as ‘T’ are expressed in bit times while parameters denoted as ‘t’ are expressed in seconds.

3.2. Main characteristics of the PROFIBUS Data Link Layer

3.2.1. General features

PROFIBUS is one of the fieldbus solutions of the General-Purpose Fieldbus Communication System European Standard – EN 50170 (1996). The PROFIBUS layered architecture is based on the ISO/OSI reference model. Only three layers are implemented: Physical Layer (PhL), Data Link Layer (referred as Fieldbus Data Link Layer – FDL – in the standard) and Application Layer (AL). Two communication profiles – PROFIBUS-FMS (Fieldbus Message Specification) and PROFIBUS-DP (Distributed Peripherals) – and several application profiles (e.g., PA – Process Automation) are supported. Every profile allows for single or multiple master systems, with a maximum of 32 stations (masters/slaves) per segment (without repeaters) and of 126 stations in one network (with repeaters).

PROFIBUS-FMS was designed for cell level applications, due to the potentiality of the application layer (reduced version of MMS – Manufacturing Message Specification). Nevertheless, in order to ensure fast and efficient data transmission, the PROFIBUS standard also provides a lighter communication profile, called PROFIBUS-DP. Its streamlined architecture uses only the physical layer and data link layer as well as a user interface, which substitutes the application layer. Typically a mono-master system, PROFIBUS-DP is a performance-optimised version of the PROFIBUS protocol and is gradually substituting PROFIBUS-FMS.

In PROFIBUS, a master can send a message on its own initiative, once it gains the right to access the bus. On the other hand, slaves do not have bus access initiative and they can only acknowledge or respond to requests from masters. Generally, slaves are peripherals such as I/O devices, valves, drives, etc. Therefore, bus access is based on a hybrid, decentralised/centralised method.

The token, that represents the right to access the bus, circulates in a logical ring composed by the masters. This token passing mechanism is based on a simplified timed token protocol (Grow, 1982), which is a well-proved solution for real-time communication systems (Agrawal *et al.*, 1994; Montuschi *et al.*, 1992; Zheng and Shin, 1995). Then, when a master station holds the token, it uses a master-slave procedure to communicate with slave stations.

PROFIBUS allows distinguishing between high priority and low priority messages. The latter can be further divided in three subtypes:

- Cyclic low priority message cycles *(*Poll Cycle), that represent the execution of the requests contained in the poll-list;
- Acyclic low-priority message cycles, which comprise application and remote management services;
- Gap maintenance cycles, that are actions taken to determine the status of the other stations in order to support dynamic changes in the network.

3.2.2. Data transfer services

The PROFIBUS FDL offers three acyclic and one cyclic data transfer services:

- Send Data with Acknowledge (SDA);
- Send Data with No acknowledge (SDN);
- Send and Request Data (SRD);
- Cyclic Send and Request Data (CSRD).

The SDA service allows an user to send data to a single remote station. If an error occurs, the data transfer is repeated. The SDN service allows an user to transfer data to a single remote station, to many remote stations (Multicast), or to all remote stations (Broadcast) at the same time, without any confirmation. The SRD service allows an user to transfer data to a single remote station and at the same time to request data from the remote station. If an error occurs, the data transfer is repeated. Finally, the CSRD service allows an user to poll remote stations (using SRD data transfers). The list of the devices to be polled is called the Poll List.

3.2.3. Message Cycle

An important PROFIBUS concept is the Message Cycle, which comprises the request frame sent by the initiator (always a master) and the associated acknowledgement or response frame from the responder (usually a slave, but can also be a master).

The acknowledgement or response must arrive before the expiration of the Slot Time, otherwise the initiator repeats the request. However, before issuing a new request, the initiator must wait a time interval defined by the Idle Time parameter. This creates the inter-frame synchronising period of idle bits each Action Frame should be preceded by. Additional detail on the Idle Time and Slot Time parameters will be presented in Sections 3.2.9 and 3.2.10, respectively.

Throughout this thesis, the terms “Message Cycle”, “Message Transaction” and “Transaction” are used interchangeably.

3.2.4. Token passing mechanism

The token is passed between masters in ascending order, up to the master with the highest address, which passes the token to the master with the lowest one. Each master knows the address of the previous station (*PS*), the address of the following station (*NS*) and its own address (*TS*).

If a master receives the token from a station that is not its previous station (*PS*), it assumes an error has occurred and does not accept the token. However, if it receives a subsequent token from the same station, it shall accept the token and assume that the logical ring has changed. In this case, it updates the originally *PS* value by the new one.

If after transmitting the token frame and after expiration of the Syn Time (T_{SYN} = 33 bits) within the Slot Time, the master receives a frame, it assumes that its successor owns the token and ceases monitoring the activity on the bus.

If the master does not recognise any bus activity within the Slot Time, it repeats the token frame and waits another Slot Time. If it recognises bus activity within the second Slot Time, it stops working as an active master assuming a correct token transmission. Otherwise, it repeats the token transmission to its next station for the last time. If after the second retry, there is no bus activity, the token transmitter tries to pass the token to the next successor. It continues repeating this procedure until it has found a successor from its list of active master stations.

3.2.5. Message dispatching

At token reception, the period during which the current master station is allowed to perform message cycles - Token Holding Time (T_{TH}) - is computed as:

$$T_{TH} = T_{TR} - T_{RR} \tag{3.1}$$

where T_{RR} - Real Rotation Time - is the time between two consecutive token receptions and T_{TR} - Target Rotation Time - is the expected time for a token cycle.

Figure 3.1 gives a clear description of the algorithm used for the message dispatching. When a master station receives the token, it may transfer at least one high priority message, even if $T_{TH} < 0$ (❶). After that, if there is Token Holding Time available, the other pending high priority message cycles are processed (❶). A master only processes other message cycles if there is still available Token Holding Time after processing all high priority messages. It should be pointed out that once a message cycle is started, it is always completed, including any retry (retries), even if $T_{TH} < 0$.

The processing of the Poll List is only started after all requested high priority message cycles have been processed (❷). After each complete Poll Cycle (say all entries of the Poll List have been processed) the requested low priority message cycles are performed in turn (❸).

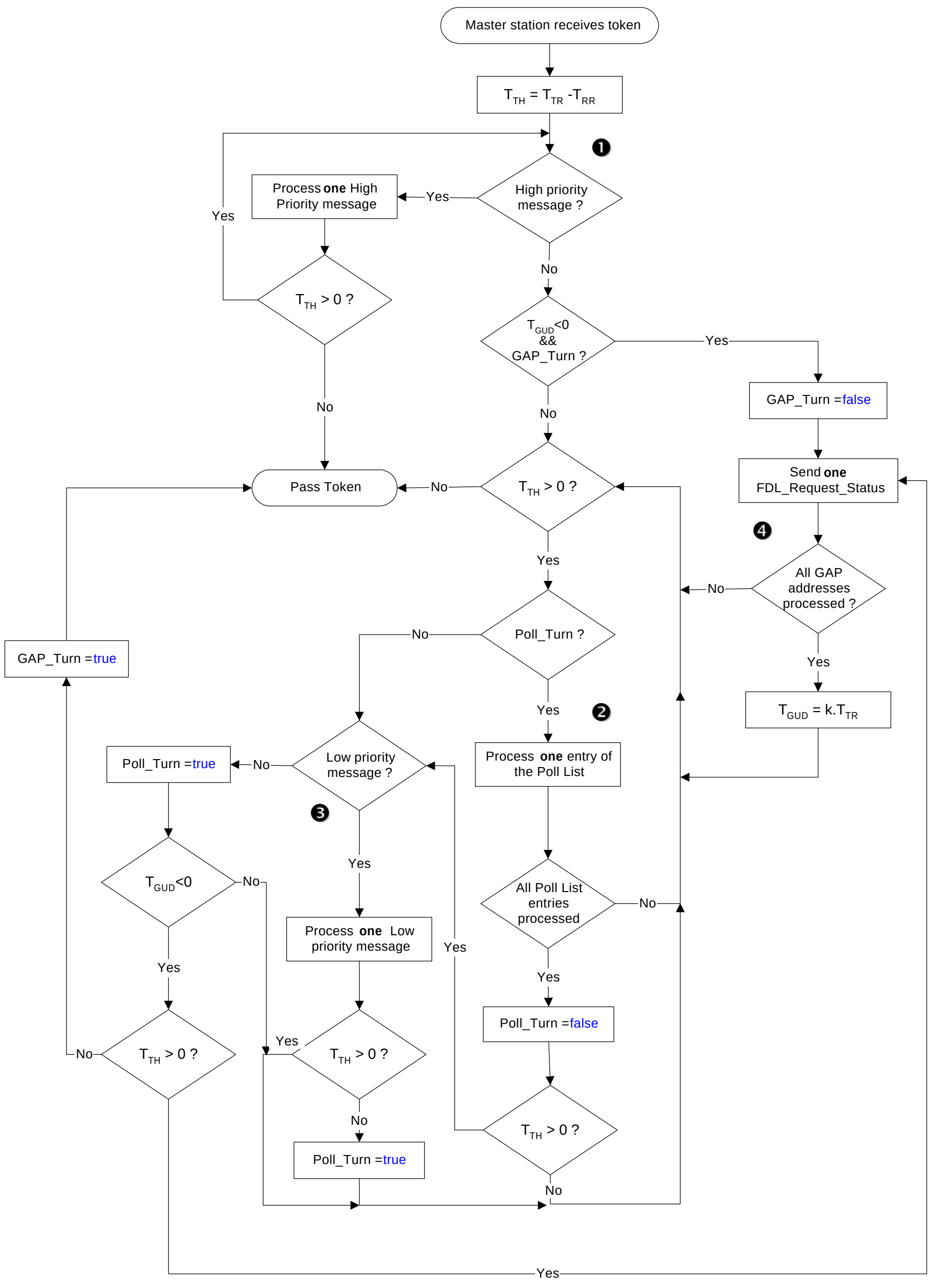


Figure 3.1: Message dispatching algorithm

The order in which low priority message cycles are performed obeys to the following rules:

- If the Poll Cycle is completed within the Token Holding Time, the low priority message cycles are carried out within the remaining Token Holding Time. A new Poll Cycle starts at the next receipt of the token;
- If at the end of the Poll Cycle there is not any further Token Holding Time available, the requested low priority message cycles are processed at the next token receipt that has Token Holding Time available for low priority message cycles. After that, a new Poll Cycle starts;
- If the Poll Cycle takes several token visits, the Poll List is Processed in segments, but without being interrupted by low priority message cycles. Low priority message cycles are performed only at the end of a complete Poll Cycle.

After all high and all low priority message cycles have been processed, if there is still time available ($T_{TH} > 0$) and if T_{GUD} has expired, the master processes one (at most one per token receipt) *GAP* update message cycle (❹). Otherwise, the GAP is updated at the next token reception, after all high priority message cycles have been processed.

3.2.6. Ring maintenance mechanisms

In order to maintain the logical ring, PROFIBUS provides a decentralised (active in every master station) ring maintenance mechanism. Each PROFIBUS master maintains two tables – the Gap List (*LGAP*) and the List of Active Stations (*LAS*) and may optionally maintain a Live List.

The Gap List consists of the address range from *TS* ('This Station' address) until *NS* ('Next Station' address, i.e., the next master in the logical ring). This includes all possible addresses, except the address range between *HSA* (Highest Station Address, that cannot be a master's address) and 127, which does not belong to the Gap List. Each master station in the logical ring starts to check its Gap addresses every time its Gap Update Timer (T_{GUD}) expires. This mechanism allows masters to track changes in the logical ring due to the addition (joining) and removal (leaving) of stations. This is accomplished by examining (at most) one Gap address per token visit, using the 'FDL_Request_Status' frame. After a complete GAP check, which may last several token rotations, T_{GUD} is reset to a multiple of the Target Rotation Time ($T_{GUD} = G \times T_{TR}$), where *G* is the Gap Update Factor, a PROFIBUS DLL parameter.

The *LAS* comprises all masters in the logical ring and is generated in each master station when it is in the *Listen Token* state, after power on. This list is also dynamically updated during operation, upon receipt of token frames. Concerning the Live List, there is the need for an explicit demand from the FDL user (above the DLL), via a management (FMA1/2) request. A 'FDL_Request_Status' frame is sent (in a cyclic way) for each destination address (0 to 126), except to the master stations, since they are already registered in the *LAS*. The correctly responding stations and the master stations in the *LAS* are entered in the Live List as existing master or slave stations.

Additionally, in order to enhance the communication system's reliability, PROFIBUS handles several error states, concerning logical ring management, such as multiple tokens (in one segment), lost token, error in token passing, multiple assignment of station addresses and stations with faulty transceivers.

3.2.7. Frame formats

In the asynchronous (RS-485) version of the PROFIBUS PhL (v1), each frame is coded in UART characters (Figure 3.2). Each UART character comprises eleven bits: one start bit (binary 0), eight data bits (octet), one (even) parity bit and one stop bit (binary 1).

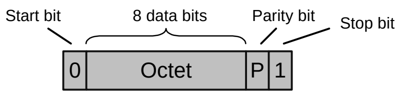


Figure 3.2: The UART character

Each *Action Frame*, that is the first frame transmitted in all transactions, must be preceded by a synchronisation period of at least 33 idle bit periods (T_{SYN}). Every frame starts with a start delimiter (SD) that characterises its type (Table 3.1).

Table 3.1: Start Delimiters and frame types

Start Delimiter	Hexadecimal Value	Frame Type
SD1	10	Fixed length frame with no data field
SD2	68	Frame with variable data field length
SD3	A2	Frame with fixed data field length
SD4	DC	Token Frame

Figure 3.3 depicts the different frame formats defined by the PROFIBUS standard, except for the short acknowledgement frame. This one comprises a single character (*SC* = E5) and is used for positive acknowledgements of SDA requests and negative acknowledgements of SRD requests.

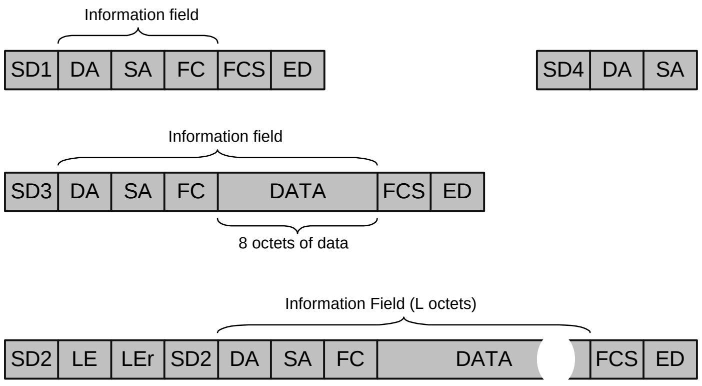


Figure 3.3: Frame formats

A fixed-length frame (request or acknowledgement) with no data field has a starting delimiter SD1, followed by the Information Field, which comprises the destination address (*DA*), the source address (*SA*) and the frame control (*FC*). The two last fields of the frame are the frame check sequence (*FCS*) and the end delimiter (*ED*), which is always the hexadecimal value 16.

Frames with fixed-length data field have starting delimiter SD3, followed by the Information Field, which also comprises a data field (*DATA*) with a fixed length of eight octets. Frames with variable data field length have start delimiter SD2 and include the length field, which is duplicated for reliability reasons (*LE=LEr*, between 4 and 249). The Information Field comprises the *DATA* field, with a length varying from 1 octet minimum to 246 octets maximum. The token frame is composed of the start delimiter (SD4) and the source and destination address fields.

3.2.8. Extended addressing scheme

The structure of the address field (both source and destination) is quite simple: the least significant bits (b_0 to b_6) represent the address of the station itself, whereas the most significant one (b_7) is the extension bit. This bit indicates the presence or absence of an extension address in the data unit after the frame control field (FC).

Both source and destination extension addresses have the same structure as the station addresses. However, the seventh bit of the extension address is no longer used to encode the address itself, indicating instead whether the remaining six bits represent a LSAP or a segment/region address. Hence, as illustrated in Figure 3.4, the Data Field consists of the (FDL or FMA1/2) User Data and an Address Field.

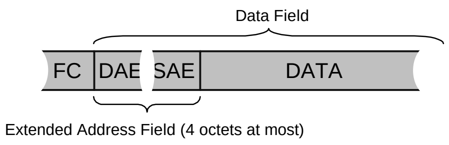


Figure 3.4: Address extensions in the Data Field

The extended addressing is applicable neither to the token frame nor to frames with no data unit. This means that only fixed and variable data length frames (starting delimiters SD2 and SD3) can be relayed through different segments of the network, according to extended addresses.

3.2.9. Additional details on the Idle Time parameters

The idle time is a period of physical medium inactivity that is inserted by master stations between consecutive message cycles. After an acknowledgement, response or token frame (Figure 3.5), a master station inserts an idle time with a value given by:

$$T_{ID1} = \max\left(T_{SYN} + T_{SM}, \min T_{SDR}, T_{SDI}\right) \tag{3.2}$$

where:

- T_{SYN} is the synchronisation time, the minimum time interval during which each station must receive idle state from the physical medium (33 bits);
- T_{SM} is a safety margin;
- min T_{SDR} is the minimum station delay of responders
- T_{SDI} is the station delay of the initiator.

Throughout this Thesis, the station delay of the responder t_{SDR} (T_{SDR} in time units) will be referred as *responder's turnaround time* – t_{rt}. This is the time elapsed since a Responder ends receiving a request PDU, until it starts transmitting the correspondent response PDU.

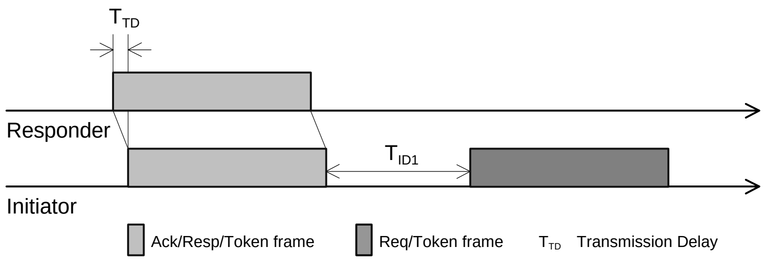


Figure 3.5: Idle Time parameter T_{ID1}

After an *unacknowledged request frame* (Figure 3.6), a master station must insert an idle time which is given by:

$$T_{ID2} = \max\left(T_{SYN} + T_{SM}, \max T_{SDR}\right) \qquad (3.3)$$

where max T_{SDR} is the maximum station delay of responders.

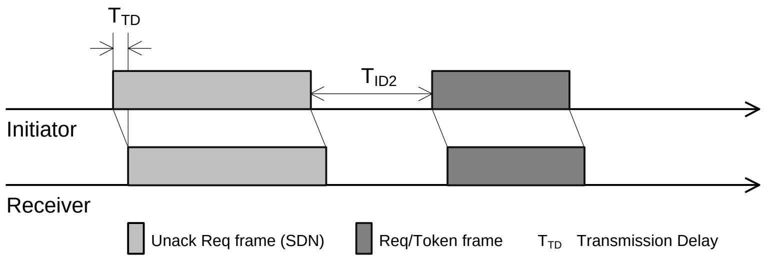


Figure 3.6: Idle Time parameter T_{ID2}

The idle time parameters can be set in a per-station basis, i.e. each master station can hold different values for the (T_{ID1}, T_{ID2}) pair. Eq. (3.2) and (3.3) are valid for a single segment network. As it will be seen later on, in a hybrid wired/wireless communication network composed of physical mediums with different bit rates and (Physical Layer) frame formats, the idle time parameters must be derived differently and at the light of a much more complex reasoning. This issue will be addressed in Chapter 6.

3.2.10. Additional details on the Slot Time parameter

The Slot Time is a parameter used by a master station to detect communication or station errors that lead to abnormal physical medium inactivity. A master station always checks if the time elapsed between the last bit of a transmitted frame and the first bit of the following frame (transmitted by another station) is smaller than the Slot Time. If this does not happen, the master station either retransmits the frame (request or token) or aborts the transmission.

An example of the use of the Slot Time parameter is illustrated in Figure 3.7, for the case of an acknowledged request frame. In the first scenario, the message transaction is completed successfully, since the response/acknowledgement frame was received within the Slot Time. On the other hand, in the second scenario, the Slot Time expired without receiving any response/acknowledge frame. In this case, the initiator either retries or aborts the message transaction, depending on the value of the *max_retry_limit* parameter (in the case of the token, the number of retries is fixed to 2, as described in Section 3.2.5).

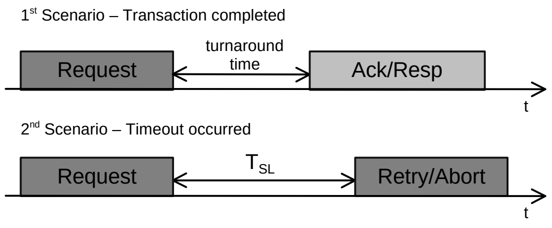

1st Scenario – Transaction completed
Request
turnaround time
Ack/Resp
t
2nd Scenario – Timeout occurred
Request
T_SL
Retry/Abort
t

Figure 3.7: Using the Slot Time parameter

In order to set the Slot Time parameter, it is necessary to compute two different components – T_{SL1} and T_{SL2}. T_{SL1} is the maximum time the initiator waits for the *complete reception of the first character of the acknowledgement/response frame*, after transmitting the last bit of the request frame (Figure 3.8).

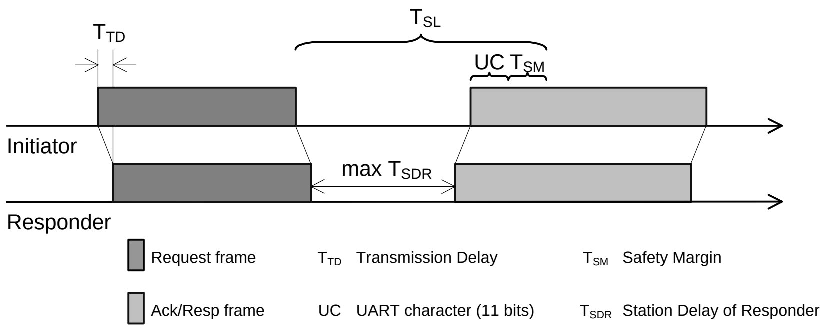


Figure 3.8: Slot Time T_{SL1}

T_{SL1} can be computed as follows:

$$T_{SL1} = 2 \cdot T_{TD} + \max T_{SDR} + 11 bit + T_{SM} \tag{3.4}$$

T_{SL2} is the maximum time the initiator waits after *having transmitted the last bit of the token frame until it detects the first bit of a frame* (either a request or the token) transmitted by the station that received the token (Figure 3.9).

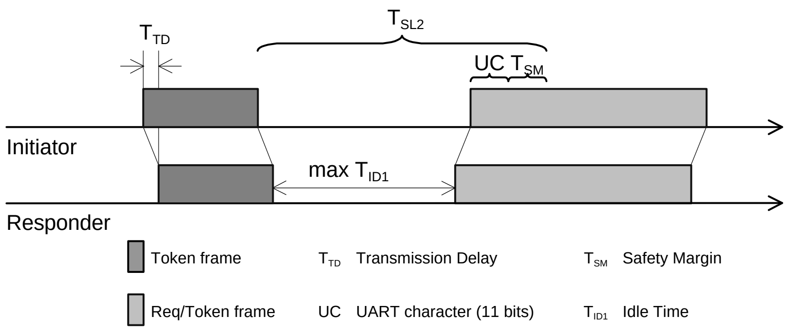


Figure 3.9: Slot Time T_{SL2}

T_{SL2} can be computed as follows:

$$T_{SL2} = 2 \cdot T_{TD} + \max T_{ID1} + 11bit + T_{SM} \tag{3.5}$$

Contrarily to the Idle Time parameters, all the master stations in the network must set the Slot Time parameter to the same value (this is imposed by the token passing mechanism), which is the maximum between T_{SL1} and T_{SL2}:

$$T_{SL} = \max(T_{SL1}, T_{SL2}) \tag{3.6}$$

While Eqs. (3.5) and (3.6) are valid for a single segment network, for a hybrid communication network composed of wired and wireless stations interconnected by repeaters, determining the appropriate T_{SL} value becomes a much more complex issue. This issue will be addressed in Chapter 7.

3.3. Real-time communication using PROFIBUS networks

Most distributed computer-controlled systems (DCCS) impose real-time requirements. In general, the issue of guaranteeing real-time requirements is the one of checking, prior to run-time, if the worst-case execution time of each of its tasks is smaller than the admissible response time. In DCCS, where some application tasks are communicating ones, the evaluation of the message's response time is of paramount importance. The message's response time is mainly dependent on the medium access delay (contention due to other messages in the queue and due to other stations holding the token), the message length and the transmission delay.

In (Vasques and Juanole, 1994) the authors provide a real-time analysis of PROFIBUS messages. However, they do not consider that PROFIBUS message requests are queued in a FCFS (First-Come-First-Served) queue. Furthermore, their analysis does not provide any estimation of the worst-case response time of each individual message.

In (Li and Stoeckli, 1996), the authors determine the maximum period for cyclic messages (CSRD). In their approach, message deadlines are guaranteed since the token cycle time is bounded. The major drawback of this approach is that, in order to evaluate the token cycle time, neither high-priority traffic nor low-priority traffic (other than cyclic traffic) is allowed. This is very restrictive in terms of using PROFIBUS to support real-time DCCS applications. Moreover, ring management traffic is not considered in their approach, since this traffic is mapped into non-cyclic low priority messages (*FDL_Request_Status* messages).

In (Tovar and Vasques, 1999a), the authors suggest two different approaches to guarantee the real-time behaviour of the synchronous traffic in the PROFIBUS networks. In the first approach – the Unconstrained Low Priority Traffic Profile, the real-time requirements for the synchronous traffic (high priority) are satisfied independently of the asynchronous (low priority) traffic load, since the protocol guarantees that at least one synchronous message is transmitted per token visit. In this way, it is possible to have a guaranteed real-time approach for the message streams provided that the relative deadline for the synchronous message streams is larger than the worst-case message response time, which is given by:

$$R_i^k = Q^k + Ch_i^k = nh^k \times T_{cycle}^k + Ch_i^k \tag{3.7}$$

where nh^k is the number of synchronous message streams generated in master *k*, T^k_{cycle} is the worst-case token rotation time and Ch_i^k is the worst-case duration of synchronous message cycle *i* issued by master *k*. The exact characterisation of the cycle time properties of the PROFIBUS token is developed in (Tovar and Vasques, 1999b), which permits the evaluation the T^k_{cycle} parameter in Eq. (3.7).

Implicit to Eq. (3.7) is the FCFS behaviour of PROFIBUS MAC message queues. Additional work can be found in the literature on how the real-time capabilities of PROFIBUS networks can be improved if priority-based strategies are implemented for serving the synchronous traffic. For instance, in (Tovar and Vasques, 2000), guaranteed approaches for both fixed priorities and deadline-based priorities are described.

The analysis in (Tovar and Vasques, 1999b) and (Tovar and Vasques, 1999a) lead to pessimistic results if applied to mono-master systems, since the authors consider always the worst-case token rotation time. Moreover, the evaluation of response time guarantees for low priority messages is not addressed. (Monforte *et al.*, 2000a) presents a new methodology for evaluating the worst-case message response time in systems where high-priority and cyclic low-priority traffic coexist in mono-master networks, where the real-time traffic is supported either by high-priority or by cyclic poll PROFIBUS messages (CSRD).

Concerning the performance of PROFIBUS, (Hong and Kim, 1997) analyse the relationship between the message delay at the DLL (simulation) and the message delay at the user layer (experimental) for PROFIBUS-FMS. (Cena *et al.*, 1997) analyse the performance characteristics of PROFIBUS-DP and CAN, taking into account parameters such as user data rate (maximum bit rate as seen by the DLL user), data coding efficiency (coding overhead of the physical and data link layers) and medium access efficiency (overhead of the MAC). For the latter, in spite of considering the overhead introduced by token-passing, they do not consider the time spent with ring management messages (GAP update *FDL_Request_Status* messages), which may be relevant for low values of the *G* factor. They conclude that, from a general point of view, the two protocols are very similar. (Benito *et al.*, 1999) extend this analysis to two other fieldbuses: Modbus Plus and PROFIBUS-FMS. For these, the performance of the cyclic services is also analysed.

(Marcos *et al.*, 2000) present a software tool called BERTA (Basic Environment for Real-Time Systems), for the schedulability analysis and simulation of distributed systems based on CAN and PROFIBUS networks. For the particular case of PROFIBUS, BERTA uses worst-case response time analysis results from (Tovar and Vasques, 1999a). (Chávez and Thomesse, 2000) analyse and compare several MAC protocols, considering both the temporal requirements of real-time applications and the characteristics of the services provided by the communication system in order to fulfil requirements such as message transfer delay, periodicity, jitter, temporal coherence and spatial coherence.

Several authors have proposed changes to the PROFIBUS MAC mechanism, in order to improve its timing behaviour. The most relevant (Vasques, 1996; Li, 1996; Tovar, 1999) consider constraining the low-priority traffic in order to avoid priority inversions in the network. This strategy increases the high-priority traffic schedulability, in a way that high-priority traffic in one station not being (so) prejudiced by low-priority traffic in other stations. (Tovar *et al.*, 2001; Ferreira *et al.*, 2001) overcome this problem by implementing a scheduling mechanism above the PROFIBUS MAC that guarantees no late tokens.

(Lo Bello and Mirabella, 2001b) presented a strategy to modify PROFIBUS token-passing policy, called the Rotating Ring, and compare its performance with the one of the standard PROFIBUS. One of the objectives is to increase fairness in the same sense as the "constrained low-priority traffic" strategy (high-priority traffic not being affected by low-priority traffic). Another objective is to reduce bandwidth waste, avoiding passing the token to stations that have nothing to transmit (lower dynamics). The Rotating Ring is composed of two virtually separated logical rings on the same physical medium, using the same token. Stations with higher transmission dynamics belong to one of the logical rings –A, in which the token rotates more frequently than in the other logical ring – B. For each token cycle in ring A (stations with higher dynamics), there is one token holding time for one station in ring B (stations with lower dynamics). The major drawback of this mechanism is that it implies changes in the PROFIBUS MAC.

Several authors have addressed the dependability characteristics of PROFIBUS. In (Willig, 1999; Willig and Wollisz, 2001), the authors investigate the behaviour of the PROFIBUS MAC over error prone physical media (e.g., wireless). They show that, for multi-master PROFIBUS networks, the protocol is very sensitive to loss/corruption of control frames (token and ring maintenance *FDL_Request_Status* frames), specially the token frame (which is not protected by a checksum). The error-detection capability of the PROFIBUS 8-bit FCS field is proved to be significantly inferior to the commonly used 32-bit checksums, in spite of the comparison not being fair (the FCS is 4 times shorter). They defend that while PROFIBUS ring stability is seriously degraded for physical layer BERs (Bit Error Rates) in the order of 10^{-2} to 10^{-3}, for BERs a magnitude smaller (10^{-4}), PROFIBUS behaviour improves significantly, since message losses are rare. Nonetheless, (Miaoudakis *et al.*, 2000) conclude that current wireless communication technologies provide a BER smaller than 10^{-5}, in typical industrial environments, while still satisfying relatively high bit rate (2 Mbit/s), a significant radio coverage (70 m) and are able to cope with high delay spreads (200 ns) and path losses (100 dB). Therefore, the PROFIBUS DLL seems to be potentially adequate for such wireless PhLs.

Besides the previously referred aspects, Willig and Wollisz suggest a number of changes to the PROFIBUS protocol, in order to improve its ring stability. The most interesting are a new method for setting timeout timers (T_{TO}) and an additional protocol feature. The new timer setting tries to prevent the breakdowns of the ring by forcing the timeout timer for current ring members to expire first, while the additional protocol feature aims at re-including lost stations as fast as possible. Importantly, stations with the modified MAC can interoperate with unchanged protocol stations, in the same network. Moreover, these improvements do not impose bandwidth or delay overhead.

In (Li, 1996), the author analyses the inaccessibility characteristics of PROFIBUS. Network inaccessibility is the time spent recovering from (usually but not necessarily) abnormal situations, i.e. time intervals when PROFIBUS does not provide regular service, although the network has not failed permanently. The author determines best and worst-case inaccessibility times for single and multiple station insertion, token loss, single and multiple station failure and inconsistency in the LAS (List of Active Stations) situations. The worst-case inaccessibility times must be added to the worst-case transmission delay (expected in the absence of faults) in order to get the worst-case response time. Additionally, the author proposes a set of rules to reduce worst-case inaccessibility times through network planning and parameterisation. The minimum HSA (Highest Station Address) policy (the HSA should be set to the lowest value possible), the address ordering policy (stations that may be subject to the same common-mode fault should have adjacent addresses) and the address assignment policy (master stations with lower addresses than slave stations). These mechanisms not only lead to a significant reduction in worst-case inaccessibility times, but also introduce no changes to the PROFIBUS protocol.

(Lo Bello and Mirabella, 1999) analyse the behaviour of PROFIBUS considering permanent and temporary faults in the physical medium (bus), in slave stations and in master stations. They also propose a Stochastic Petri Net model that permits to analyse the fault tolerance behaviour of PROFIBUS and present some numerical results resulting from a simulation. (Carvalho and Portugal, 2001) present a framework to model and evaluate the dependability of fieldbus networks in the presence of faults, through Markov Chains. Several architectures are analysed, considering different operating modes, topologies and fault types. Nevertheless, since the abstraction level is considerably high, further work is necessary to adapt these models to each particular fieldbus network.

Another issue that has been gaining increasing attention is the support of multimedia traffic over PROFIBUS networks. Within the framework of the IST RFieldbus project, industrial multimedia TCP/IP applications are supposed to coexist with PROFIBUS-DP control applications. The integration of TCP/IP within the PROFIBUS protocol stack must provide an adequate Quality of Service to the supported TCP/IP applications, while guaranteeing that the timing requirements of the control-related traffic (“native” PROFIBUS traffic) are always satisfied. Such integration is achieved through three sub-layers: IP-Mapper, Admission Control and Scheduler and Dispatcher (Pacheco *et al.*, 2001; Tovar *et al.*, 2001; Ferreira *et al.*, 2001). A more limited solution is proposed by (Sempere *et al.*, 2001; Blanes *et al.*, 2001), which encompasses an “image transport system” over PROFIBUS-DP networks. The proposed architecture neither supports standard TCP/IP applications nor the coexistence of multimedia and PROFIBUS-DP applications in the same network station.

Chapter 4

Principles and Design Rules for the System Architecture

This chapter discusses the architecture for a wired/wireless PROFIBUS-based network. First, the different components of such a network are characterised. Then, some major design alternatives are presented, justifying the choice for an approach where the Intermediate Systems behave as repeaters. In addition to this, some design rules that govern the proposed approach are introduced in detail. Finally, an innovative mechanism that is able to support inter-cell mobility with the desired requirements is described.

4.1. Introduction

This chapter focuses on the major design alternatives for a hybrid wired/wireless PROFIBUS-based network. It starts by defining the major components of the Communication Network: Wired and Wireless End Systems, Structuring, Linking and Structuring & Linking Intermediate Systems, Wired Domains, and Ad-hoc and Structured Wireless Domains (Section 4.2). Then, some examples of possible Communication Network topologies are presented, together with the definition of some basic interoperability rules.

At this point, PROFIBUS-DP is assumed as the user interface on both wired and wireless End Systems (ESs). It is also assumed that wired ESs use PROFIBUS Data Link and Physical Layers. The Data Link and Physical Layers of wireless ESs and ISs are initially open. Given that even within this more restrict scope (PROFIBUS-based network), different design alternatives for the hybrid wired/wireless network architecture can be devised.

The behaviour of the Intermediate Systems operating at Physical, Data Link and Application Layers is briefly analysed, to the point where the option for a Layer 1 Intermediate System is sustained. Some additional interoperability rules for the adopted repeater-based approach are also presented and a simple and time bounded mobility management mechanism is briefly described.

In the remainder of this thesis, the term “PDU” (Protocol Data Unit) will be used interchangeably with the term “frame” (previously used in Chapters 2 and 3) and the terms DLL PDU (Data Link Layer PDU) and PhL PDU (Physical Layer PDU) will be used to refer PDUs at the DLL and PhL, respectively.

4.2. Network components

Throughout this thesis, the hybrid wired/wireless fieldbus network will be simply referred to as the “Communication Network”. In this section, the different components of such a Communication Network are described.

4.2.1. End Systems and Intermediate Systems

A Communication Network (Figure 4.1) is basically constituted by devices that support end-user applications/services and network interfaces – End Systems (ESs) – and by devices that are used for network interconnection – Intermediate Systems (ISs). The End Systems can have a wired or wireless interface, i.e., they can be Wired End Systems (WRESs) or Wireless End Systems (WLESs).

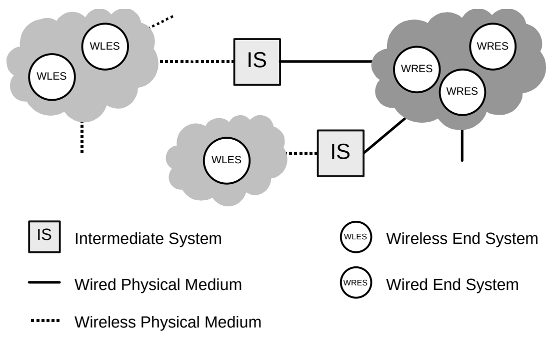


Figure 4.1: Example of a hybrid wired/wireless communication network

4.2.2 Wired and Wireless Communication Domains

A set of End Systems and Intermediate Systems that communicate directly via a wired physical medium is called a Wired Communication Domain, or Wired Domain (WRD), for short. A Communication Network composed exclusively of Wired Domains is called Wired Network.

Wireless ESs have a wireless network interface enabling the communication while moving within a pre-defined three dimensional region. Depending on the dimension and layout of this radio coverage region, on the existence of electromagnetic interference and obstacles and on the radio technology that is being used, there may be the need to split the radio coverage region into a number of smaller regions (radio cells).

A Radio Cell (RC) is therefore defined as a common radio coverage area of a group of WLESs and ISs. The set of WLESs/ISs that defines a Radio Cell is called Wireless Communication Domain, or Wireless Domain (WLD), for short. A Radio Cell can be either ad-hoc (ARC) or structured (SRC), depending on whether the WLESs/ISs communicate directly or indirectly between them. Structured Radio Cells are necessary when inter-cell mobility must be supported, i.e. when End Systems must be able to communicate while moving from one Radio Cell to another. This requirement will become clearer later on in this chapter. The mechanism that supports inter-cell mobility is called handoff (or handover). In order to have a Structured Radio Cell, there is the need for a specific type of IS – a Structuring Intermediate System (SIS). A Structured Wireless Domain (SWLD) is defined as the set of End Systems and Intermediate Systems that are associated to a Structured Radio Cell.

All communications between ESs belonging to a Structured Wireless Domain must be relayed by the Structuring Intermediate System (Figure 4.2). ESs transmit using one Radio Channel (uplink) and receive using another Radio Channel (downlink). A Radio Channel Set (CHS) is defined as the set of radio channels used for communication within a Radio Cell. Therefore, in a Structured Radio Cell, the Radio Channel Set is composed of two Radio Channels, one uplink and one downlink. Structuring Intermediate Systems may implement half or full-duplex communication.

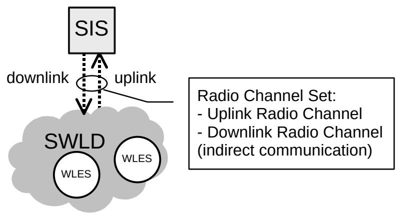


Figure 4.2: Structured Wireless Domain (SWLD)

If mobility is to be performed inside a specific Radio Cell (intra-cell mobility), then Ad-hoc Radio Cells can be used. An Ad-hoc Wireless Domain (AWLD) is defined as a WLD where associated ESs define an Ad-hoc Radio Cell (Figure 4.3). In this case, it is assumed that the Radio Channel Set is composed of only one Radio Channel, which is used both to transmit and to receive, using half-duplex communication.

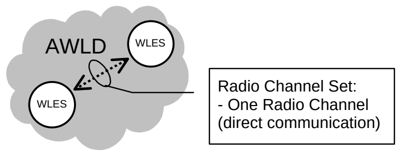


Figure 4.3: Ad-hoc Wireless Domain (AWLD)

An application may impose or benefit from the support of Mobile Wired Domains – MWRDs (Figure 4.6). As an example, consider an automatic vehicle (AGV) containing a Wired Domain with a set of wired ESs, moving between Radio Cells. All the End Systems associated to the MWRD move altogether and the (one and only one) IS that is associated to the MWRD (a mobile LIS, as described in the next section) must handoff between Radio Cells. This is described in more detail in Section 4.3.4.

4.2.3. Intermediate Systems functionality

Three basic types of Intermediate Systems (ISs) are considered: the Linking Intermediate System (LIS), the Structuring Intermediate System (SIS) and the Structuring & Linking Intermediate System (SLIS), which is a combination of the first two types. Additionally, the mobility feature of mobile Wired Domains imposes the definition of a fourth type of IS – the Mobile Linking Intermediate System (MLIS)

While ISs may be used to interconnect more than two Communication Domains (e.g. hubs, switches, Internet routers), it is assumed that all the four types, described next in more detail, interconnect two Communication Domains only. This simplification does not restrict any of the mechanisms and methodologies proposed in this Thesis, which also apply if the more general case was considered.

Linking Intermediate System (LIS)

A Linking Intermediate System (LIS) (Figure 4.4) interconnects a Wireless Domain (WLD) and a Wired Domain (WRD). The LIS can be associated to an Ad-hoc Wireless Domain (AWLD) or to a Structured Wireless Domain (SWLD), depending on whether the Radio Cell is ad-hoc or structured, respectively. PDUs arriving from the WLD are relayed to the WRD and vice-versa. Obviously, if the LIS performs routing (discussed later on in this chapter), then only a subset of the incoming PDUs is relayed.

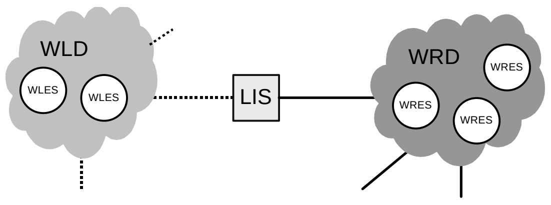


Figure 4.4: Linking Intermediate System (LIS)

Structuring Intermediate System (SIS)

A Structuring Intermediate System (SIS) interconnects Wireless End Systems (WLESs) and Linking Intermediate Systems (LISs) associated to a Structured Wireless Domain (SWLD). Since the SIS creates a Structured Radio Cell, all WLESs/LISs associated to the Structured Wireless Domain communicate with one another via the SIS (Figure 4.2). PDUs arriving from the SWLD in the uplink Radio Channel are retransmitted through the downlink Radio Channel.

Structuring & Linking Intermediate System (SLIS)

A Structuring & Linking Intermediate System (SLIS) combines both LIS and SIS functionalities in a single device (Figure 4.5).

A SLIS creates a Structured Radio Cell. Therefore, PDUs arriving from the Structured Wireless Domain (SWLD) in the uplink Radio Channel are retransmitted in the downlink Radio Channel. Moreover, PDUs arriving from the SWLD (uplink Radio Channel) are relayed to the WRD and PDUs arriving from the WRD are relayed to the

SWLD (downlink Radio Channel). Similarly to the LIS, if the SLIS performs routing, then only a subset of the incoming PDUs is relayed.

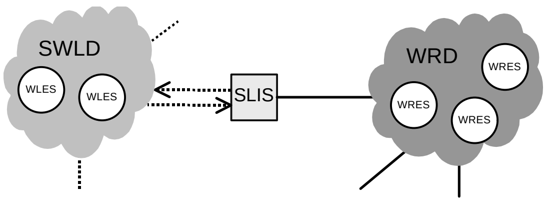


Figure 4.5: Structuring & Linking Intermediate System (SLIS)

When compared to a SIS, a SLIS has the advantage of reducing communication latency, when relaying PDUs between Wired and Wireless Domains. In fact, when relaying from wired to wireless, the uplink radio transmission is not used. In turn, when relaying from wireless to wired, the downlink radio transmission is not used. The disadvantage is that a wire must reach the SLIS, i.e. cabling problems may emerge.

Mobile Linking Intermediate System (MLIS)

If mobility of Wired Domains must be supported, there is the need for a Linking Intermediate System with mobility capabilities – the Mobile Linking Intermediate System (MLIS). Figure 4.6 depicts a Mobile Wired Domain (MWRD), associated to a MLIS:

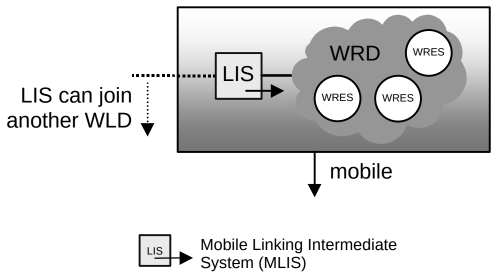


Figure 4.6: Mobile WRD and mobile LIS

As a side remark, a fifth type of IS (previously only four types were mentioned) could also be considered, which is one able to interconnect two Wired Domains. Nevertheless, that type of IS would not introduce any important issues to the architectural discussion. Additionally, it should be stressed that all the mechanisms and methodologies proposed in this Thesis would also apply for this particular type of IS.

4.2.4. Basic interoperability rules

The above-defined components allow different Communication Network topologies, namely a “pure” Wireless, a “pure” Wired and, generically, any Hybrid Wired/Wireless Communication Network topology.

We introduce now some rules, and their rationale, that must be observed to guarantee proper operation of the network and interoperability between network components.

- A Communication Network is composed by at least one Communication Domain (WRD or WLD);
- A Mobile Wired Domain is associated with one and only one Mobile Linking Intermediate System;
- If more than one Linking Intermediate System is associated to the same Wired Domain, each of them must be associated to a different Wireless Domain (to avoid closed loops);
- A Wireless Domain can be associated with at most one IS of structuring type (SIS or SLIS) (see the example depicted in Figure 4.7a);
- The support of inter-cell mobility requires at least two Structured Wireless Domains with overlapping Radio Cells (example depicted in Figure 4.7b).

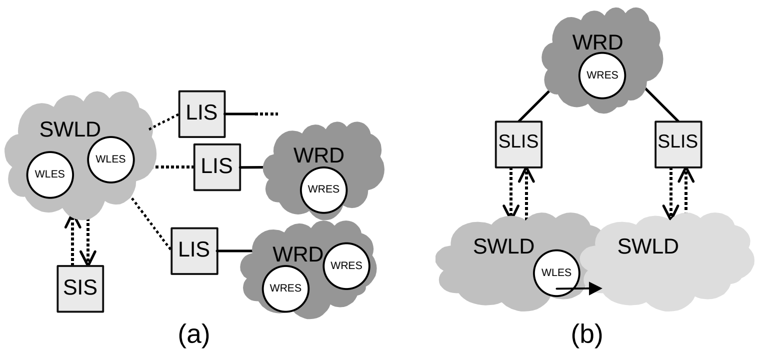


Figure 4.7: Examples of network topologies

Annex A presents an object-oriented model of the Communication Network. This model represents all the network components (as objects) and considers the above mentioned rules.

4.3. Analysis of design alternatives

This section analyses a number of different design approaches for the Communication Network architecture. It particularly focuses on the rationale for choosing Intermediate Systems operating at the Physical Layer level.

As previously mentioned Section 4.1, PROFIBUS-DP is assumed as the user interface on both Wired and Wireless End Systems. It is also assumed that Wired ESs use PROFIBUS Data Link and Physical Layers. The Data Link and Physical Layers of Wireless ESs and ISs are going to be outlined in the rest of this section.

4.3.1. Intermediate Systems behaviour

The behaviour of the ISs, which theoretically speaking may act as repeaters, bridges, routers or gateways, is an important aspect that must be analysed. Repeaters, bridges, routers and gateways relay data at Physical Layer (PhL), Data Link Layer (DLL), Network Layer (NL) or above NL, respectively. Theoretically, the higher the OSI layer at which the Intermediate System operates, the larger the communication latencies may result. This aspect impacts on the duration of message transactions, which is a very important aspect in a Communication Network based in PROFIBUS-DP; that is, the time spent sending a request PDU and receiving the related response PDU.

Within this context, the PROFIBUS Slot Time parameter – T_{SL} (refer to Chapter 3) assumes a particular importance. On one hand, T_{SL} must be set large enough to cope with extra latencies introduced by the Intermediate Systems. On the other hand, T_{SL} must be set as small as possible, in order to guarantee an acceptable level for the responsiveness to failures; that is, a master must detect a message/token loss or a station failure within an acceptable time interval. Most importantly, since T_{SL} is a time component of the worst-case duration of a message transaction, its value will impact the evaluation of the worst-case message response time.

The different options for the ISs are going to be described next.

Repeater operation

Traditionally functioning just as a signal regenerator, a repeater can also interconnect two networks with different Physical Layer protocols (e.g. different bit rates). Layers 2-7 protocols must be the same for both Communication Domains.

In the context of this thesis, a repeater is classified according to its relaying behaviour: *store-and-forward*, where a PhL PDU must be completely received from one port before being transmitted to the other port; *cut-through*, when the IS starts relaying a PhL PDU which has not been completely received yet.

A repeater may need to implement more than a bit-by-bit repeating functionality. This is the case when it interconnects communication media with different PhL PDU formats. Another example of additional functionality is when a repeater has to support encryption/decryption for security reasons.

A repeater does not perform any address filtering. This results in a broadcast network, i.e. every ES listens to every PDU transmitted by any other ES in the Communication Network. The use of repeaters implies a single MAC address space and a single logical ring in the Communication Network. A MAC address space is a set of ESs (and ISs, if these are not repeaters) with unique MAC addresses. A logical ring is a set of ESs (and ISs, if these are not repeaters) that take part in a single token passing procedure.

When a PROFIBUS master ES sends an acknowledged message request that must be relayed by several ISs to reach a PROFIBUS slave ES, the duration of the message transaction depends on the relaying latency of the repeaters. Obviously, cut-through repeaters introduce lower relaying latencies than store-and-forward repeaters. Nevertheless, the latter may advantageously implement error correction functionalities.

Bridge operation

A bridge is able to interconnect two Communication Domains with different DLL (and PhL) protocols. Layers 3-7 protocols must be the same for both Communication Domains. There are several classification criteria in the literature, such as the one proposed by (Varghese *et al.*, 1990). Here, we simply focus on its routing and contention characteristics.

Usually, a bridge performs routing functionality according to DLL (MAC) addresses. It keeps two dynamic address tables used to control frame relaying from one port to the other . Nevertheless, (Kunert, 1997) introduces the concept of "transparent bridge", where all the incoming PDUs are relayed, regardless of the DLL destination address (no-routing functionality). This is a kind of "layer 2 repeater", since it still may perform some DLL functionalities (e.g. changing DLL PDU format). This is a relevant aspect in the context of this section and therefore the issue whether the bridge performs routing or not will be further analysed here.

Bridges can also differ depending on whether they contend or not for medium access (with the ESs). Therefore, contention and no-contention bridges will be briefly discussed. This characteristic applies to both Communication Domains the bridge is associated with. Although this characteristic usually applies to both Communication Domains, in some cases a bridge may not contend for medium access in one of its ports and contend for medium access in the other port (Kunert, 1997).

Figure 4.8 illustrates an example where an IS acting as a contention type bridge interconnects two Communication Domains (one wired and one wireless) with PROFIBUS MAC. This is an example where there is contention for medium access on both sides. Consequently, both sides of the ISs have master functionality, resulting in two separate logical rings.

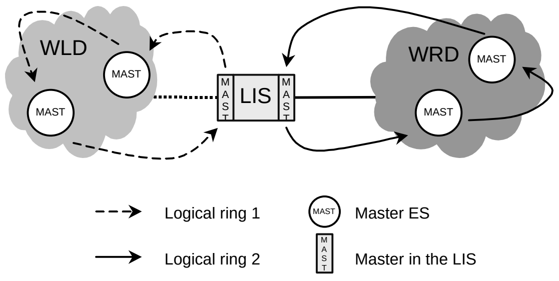


Figure 4.8: Intermediate System acting as a bridge

Bridges that contend for medium access in Communication Domains using the PROFIBUS MAC can potentially have a synchronous or asynchronous behaviour, depending on whether the initiator of a transaction waits for the responder's response or it receives an immediate acknowledgement to a request from the first IS, respectively. We will show now that none of these approaches are possible.

In the asynchronous approach, when the bridge receives a request PDU that must be relayed to the other port, it should send an immediate acknowledgement to the initiator informing that "response is due later". PROFIBUS does not support handling this type of PDUs, so this solution is not possible.

In the synchronous approach, the bridge just relays the PDU, without acknowledging the sender. This is also not possible with the PROFIBUS protocol. In fact, consider the Communication Network illustrated in Figure 4.9. Given that each Communication Domain (WLD and WRD) has its own logical ring, there is the possibility of both token holders (WLES1 and WRES2) issuing (almost) simultaneous request PDUs to responders in the other Communication Domain (WRES1 and WLES2, respectively), i.e. request PDUs that must be relayed by the bridge. In this case, the bridge enters in a deadlock state, since it has no medium access neither to relay the Request PDU from WLES1 to WRES1, nor from WRES2 to WLES2. Therefore, contention bridges cannot be used in PROFIBUS-based networks.

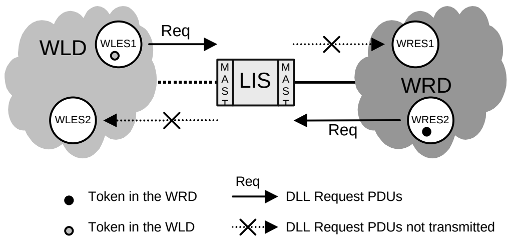


Figure 4.9: Deadlock situation in a synchronous bridge

In (Ferreira *et al.*, 2002), a (slight) modification to the PROFIBUS DLL is proposed, enabling the support of asynchronous transactions (based on the use of an immediate “response due later” type of PDU).

The relaying latencies introduced by no-contention bridges are higher than those introduced by repeaters, as DLL PDUs must be translated from one protocol to the other (even if just a “tunnelling” is performed). Nevertheless, a bridge enables the use of a different DLL protocol in wireless communications, as it is discussed in Section 4.3.2.

Router operation

A router is capable of relaying PDUs according to their Network Layer addresses. In multiple path (hop) networks, the router can also be responsible for controlling/avoiding traffic congestion and for guaranteeing fair routing decisions, based on knowledge of the topology and conditions of the communication network. Layers 4-7 protocols must be the same for both Communication Domains.

Since PROFIBUS does not have a Network Layer, the option for a router is ruled out. However, it is important to point out a common misconception about PROFIBUS extended addresses. PROFIBUS extended addressing scheme permits to implement a kind of network addressing. ESs/ISs associated to the same Communication Domain could share the same extended address. Therefore, routing could be implemented, knowing the extended addresses of every ES. There are two problems, though. First, only fixed and variable data field PDUs include extended addresses. This means that only these could be routed through the ISs. Second, this “network address” loses its interest when inter-cell mobility is supported. This is so since a mobile ES can move from one Wireless Domain to another, but its extended (network) address continues to be the same. As a consequence, router type of ISs are not considered for the architecture in discussion.

Gateway operation

A gateway is an IS that performs relay functionality above the Network Layer. Two Communication Domains with completely different 1-7 layer protocols may be interconnected using a gateway. However, most of the times (always, in fieldbuses), gateways perform interconnection on top of the Application Layer (AL). This can be achieved in at least two different ways (Decotignie *et al.*, 2001), as described next.

Gateways with synchronous behaviour ("protocol converter") receive an application service indication from one Communication Domain and convert it in an application service request on the other Communication Domain. When the corresponding confirmation is received, it converts it in a reply on the other side.

In order to overcome the resulting higher latencies in request/response transactions, some gateways maintain an updated image of process data by issuing requests to the mirrored communication network, asynchronously ("proxy-like" behaviour). When a read request is received from one Communication Domain, the cached information (from the other Communication Domain) is returned in the response. When a write request arrives to the proxy gateway from one Communication Domain, the image is updated and a write request is asynchronously issued to the other Communication Domain. The proxy-like gateway is the most common approach in commercial products (Sink, 2000), for the interconnection of dissimilar fieldbus networks.

With the PROFIBUS protocol, synchronous gateways can enter in deadlock state just like synchronous bridges. Contrarily, proxy-like gateways may be used and lead to very short turnaround times, since the response/acknowledgement is immediately issued (no relaying). Nevertheless, the functionality of an asynchronous gateway is limited, due to the complexity to maintain a truthful image of process data. This is particularly true in case the Communication Network contains a considerable number of ESs, ISs and, most importantly, if inter-cell mobility must be supported. In fact, the mobility of ESs (MWLESs) or Wired Domains (MWRDs), would turn extremely difficult to keep their image in every (proxy) gateway updated.

Therefore, proxy-like gateways are not potential candidates for interconnecting Communication Networks with complex topologies and considerable information complexity. This kind of gateways can however be adequate for Communication Networks with just two Communication Domains.

4.3.2. Communication protocols for the Wired and Wireless Domains

In a generic hybrid wired/wireless communication network, there are multiple options for the communication protocols. In the proposed architecture, the application layer (user interface) protocol is considered to be PROFIBUS-DP, for both Wired and Wireless Domains and Wired Domains use the RS-485 PhL (v1) of PROFIBUS (Figure 4.10).

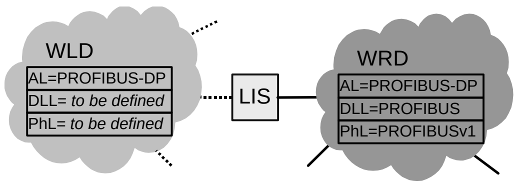


Figure 4.10: PROFIBUS-DP as the AL protocol

Several design alternatives concerning the PhL and DLL of Wireless Domains are considered next.

Physical Layer (PhL) for the WLDs

It is assumed that Wired Domains use the RS-485 asynchronous version of the PhL, due to the high bit rates supported. Nevertheless, a different PhL protocol is required for the Wireless Domains. This may be based on one of the wireless PhL communication protocols that were mentioned in Section 2.2.4. One of the most prominent candidates is IEEE 802.11b, since it has technical features that fulfil the envisaged requirements and the associated technology is mature.

Data Link Layer (DLL) for the WLDs

Concerning the Data Link layer, two alternatives would be possible, depending on whether the Wireless ESs (WLESs) are supposed to have a RS-485 wired PhL or not.

If only "traditional" RS-485 PROFIBUS boards were to be used, then a different DLL protocol for the Wireless Domains could be considered. In this case, ISs could not work as repeaters. Each WLES would create its own Wired Domain, requiring its own LIS, in order to interconnect that Wired Domain (PROFIBUS DLL/PhL protocol) and the Wireless Domain (wireless DLL/PhL protocol). This is illustrated in Figure 4.11.

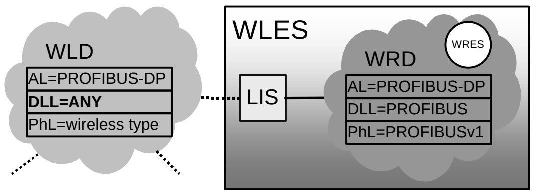


Figure 4.11: WLES with "traditional" PROFIBUS board

Examples of candidate wireless network protocols for the DLL of Wireless Domains are standards such as UMTS, IEEE802.11 and Bluetooth. Such a solution would eventually require the use of no-contention (and no-routing) bridges ("layer 2 repeater") performing "tunnelling" (encapsulation of a PROFIBUS DLL PDU in the data field of a wireless protocol DLL PDU).

Contrarily, considering that WLESs have a specific wireless PhL (non-PROFIBUS), then wired and wireless ESs must have the same DLL protocol. Therefore, the wireless MAC must (also) be the PROFIBUS MAC (Figure 4.12). In this situation, the Intermediate Systems can operate at the Physical Layer (as repeaters), since only the Physical Layer protocols are different.

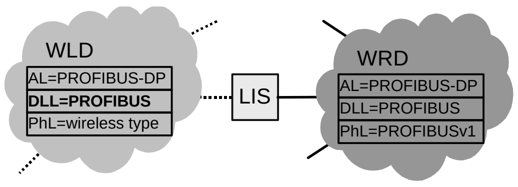


Figure 4.12: WLES with specific wireless PhL

4.3.3. Support of intra/inter-cell mobility

The support of intra-cell mobility in a Communication Network is quite trivial, since there are no changes in the WLDs, i.e., ESs/ISs neither join nor leave WLDs. Therefore, dynamic routing mechanisms are not needed. Assuming that the ISs act as repeaters, the Communication Network is composed of a single address space and a single logical ring (refer to Section 4.3.1). In this case, Radio Cells may operate in the same Radio Channel Set, provided that they do not overlap. While the PROFIBUS MAC guarantees that there is only one node transmitting at a given moment (in the overall Communication Network, since there is a single logical ring), if two Radio Cells operating in the same Radio Channel Set overlap, duplicated or missing PDUs may occur.

The support of inter-cell mobility means that ESs/ISs may join and leave WLDs, dynamically. Adjacent Radio Cells must operate in different Radio Channel Sets and must overlap, in order to provide continuous connectivity to mobile ESs/LISs.

4.4. An architecture for the Communication Network

At the light of the analysis outlined in the previous sections, the Communication Network is assumed to have ISs acting as repeaters, i.e. the wireless parts can be seen as simple extensions of the wired parts. Wired Domains use the PROFIBUS PhL v1, while a PhL protocol similar to the one defined in the IEEE802.11b standard is assumed for the Wireless Domains. As it was explained in Section 4.3.2, since wireless ESs with a specific wireless PhL (non-PROFIBUS) are to be supported, wired and wireless ESs are required to have the same DLL protocol. Therefore, the wireless MAC must (also) be the PROFIBUS MAC.

ISs with structuring functionality (SISs and SLISs) should always behave as repeaters, when relaying PhL PDUs within (Structured) Wireless Domains. In fact, every transmission must be broadcast to all ESs/LISs within a SWD. Consequently, there would be no need for ISs to operate above the Physical Layer.

A broadcast network based on repeaters permits to implement a handoff mechanism based on Radio Channel assessment and switching, as it is described in Section 4.4.2. However, it should be noted that the use of repeaters in such a hybrid architecture still introduces additional communication latencies (when compared to the pure wired solution) that must be taken into consideration. This issue will be addressed in Chapters 6 and 7.

4.4.1 Additional interoperability rules

For the proposed approach, where the overall Communication Network behaves as a "broadcast" network (ISs act as repeaters), there is the need to introduce some additional rules that add to those introduced in Section 4.2.4.

In order to have a proper behaviour of the network there can be no more than one possible path between any two Communication Domains (tree topology), i.e. no closed loops can exist. In order to support inter-cell mobility, Radio Cells must overlap (to provide continuous connectivity to mobile ESs/LISs). Due to the broadcast nature of a network based on repeaters, overlapping Radio Cells must operate in different Radio Channel Sets, to avoid duplicated (collisions) or missing PDUs.

Consider the example depicted in Figure 4.13. Two LISs associated to the same WRD receive a transmission from a WLES (located in the overlapping region) and relay the signal into the WRD, causing a collision.

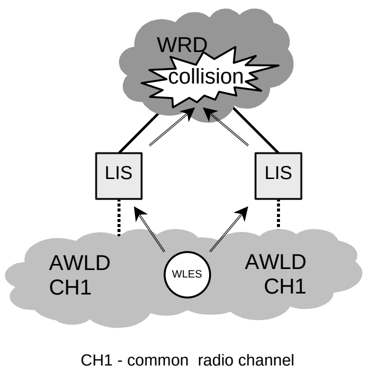


Figure 4.13: Collision caused by common Radio Channel Set

Since the number of Radio Channels available is limited, it may happen that a WLES receives simultaneous transmissions from both its own Communication Network and another (neighbour) Communication Network, resulting in jammed communication.

In order to detect whether a received PDU was transmitted from an ES belonging to the same Communication Network or not, some kind of Communication Network Identifier should be included in the wireless PhL PDU (e.g. in the header). This identifier is needed since each Communication Network holds a separate MAC address space (thus two ESs belonging to different Communication Networks can have the same MAC address). Using this identifier, if a wireless End System receives a PDU from an End System belonging to another Communication Network, it just discards it.

Nevertheless, the problem of wireless jamming can only be avoided if at the system design phase it is guaranteed that all overlapping Radio Cells belonging to neighbour Communication Networks operate in different Radio Channel Sets. This is a typical (radio) system-planning task, which is out of the scope of this Thesis.

4.4.2. Supporting inter-cell mobility

A Communication Network based solely on Linking Intermediate Systems leads to Ad-hoc Wireless Domains. These have the advantage of permitting direct communication between wireless ESs/LISs. Nonetheless, power consumption restrictions inherent to mobile devices may lead to insufficient radio coverage to fulfil the system requirements (a WLES/LIS not being able to reach all the wireless ESs/LISs in its (Ad-hoc) Wireless Domain). Moreover, factory environments quite often have harsh electromagnetic interference and barriers that increase this problem. Finally, if mobility of Wired Domains was to be supported, there would be the need for a procedure to guarantee that LISs belonging to mobile Wired Domains would switch between Radio Channel Sets.

Taking all these requirements into account, a structured radio environment based on Structuring and/or Structuring & Linking Intermediate Systems is the most adequate. SIS/SLIS devices can be fixed and fed by the power network, leading to an increased reception/transmission capacity, and can be located in strategic places, turning the problem of hidden nodes (caused by electromagnetic barriers) easier to tackle.

The mobility management mechanism assumed for the approached architecture was developed within the RFieldbus Project (IST-1999-11316). It was firstly described in an project internal document (Koulamas *et al.*, 2001b) and then published in (Alves *et al.*, 2002). This mechanism provides a seamless handoff for all kinds of mobile ESs (master/slave) and mobile LISs (associated to MWRD). Due to the broadcast nature of the Communication Network, the proposed mobility management mechanism just encompasses a procedure for Radio Channel assessment and switching. Importantly, the proposed mobility management mechanism guarantees that there is no PDU loss (considering no faults) and permits to fulfil stringent real-time requirements. In fact, mobility management is restricted to a reduced, well-determined and bounded period of time (as it will be shown in Chapter 8). The basics of this mechanism are outlined next.

One specific station - the *mobility master (MobM)* - must support some additional functionality, since it is responsible for triggering the mobility management procedure (Figure 4.14). Within a certain period – the *beacon period*, all mobile ESs/LISs are expected to assess the quality of the different Radio Channel Sets (CHS1-3), finally switching to the one considered as having the best quality. After this period for the mobility management procedure, the MobM is able to pass the token to another master.

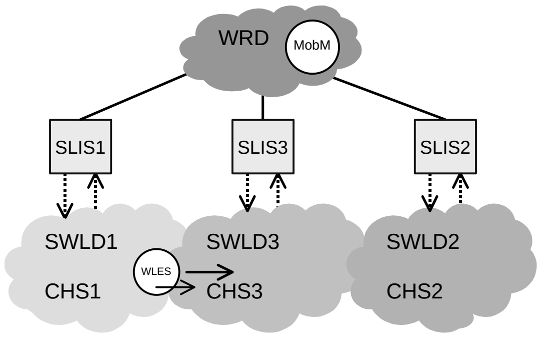


Figure 4.14: The mobility master (MobM)

The mobility master (i.e., the master that has the responsibility of triggering the handoff procedure) sends a special (unacknowledged) PDU – the *beacon trigger* (BT), with a periodicity that is dependent on the maximum speed of the mobile stations (Koulamas *et al.*, 2001b). This BT PDU is broadcast to the entire network and causes each SIS/SLIS to send a number of beacons in its downlink Radio Channel. Mobile ESs/LISs receive these beacons, assess the signal quality of all (downlink) Radio Channels and switch to the Radio Channel Set with best quality (Figure 4.15).

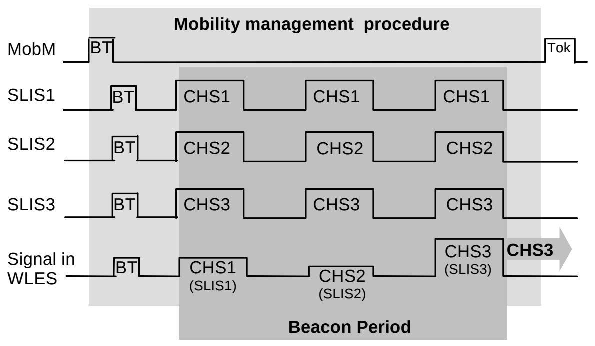


Figure 4.15: Mobility management procedure timing diagram

For the scenario depicted in Figure 4.14 where the mobile ES is moving towards the range of SLIS3, the (M)WLES assesses the channel and switches to CHS3 (Figure 4.15).

4.5. Summary

This chapter discussed relevant issues on the architecture of a hybrid wired/wireless network based on the PROFIBUS protocol. A number of generic components for building such a network were defined, namely several types of devices for network interconnection (Intermediate Systems) were characterised and a set of rules established.

Within this context, Intermediate Systems operating at Physical, Data Link and Application Layers was briefly analysed, to an extent sufficient to sustain the choice for a repeater-based approach. This results in a broadcast network, where inter-cell mobility can be supported by guaranteeing that Radio Cells overlap and that adjacent Radio Cells operate in different Radio Channel Sets. Mobility management is then reduced to the mobile ESs/LISs assessing the quality of the different Radio Channel sets existent in the network (SISs/SLISs transmit beacons in their downlink Radio Channel, during well-defined periods), and switching to best quality one. This mechanism copes with transparency, simplicity and, as it will be demonstrated in Chapter 8, with real-time requirements.

Chapter 5

Analytical Models for the Communication Network

In this chapter, analytical models for the Communication Network are proposed. These models essentially cover the Communication Domains, Intermediate Systems and Physical Media. Their timing characteristics are defined, thus enabling the appropriate timing analysis carried out in Chapters 6, 7 and 8.

5.1. Introduction

The addressed Communication Network is composed of a number of different components, namely Communication Domains, Intermediate Systems and End Systems. In this chapter, an analytical model for such Communication Network is proposed, including the definition of the parameters and behaviours relevant to the scope of this thesis.

First, the components of the Communication Network are grouped into a number of sets, organised according to the type of component. Then, the attributes of Communication Domains and End Systems are defined and some additional conditions for supporting mobile End Systems (MWLESs) and mobile Wired Domains (MWRDs) are presented. After, the model of the physical media is defined, including the format of a Physical Layer PDU. Such aspects are of paramount importance in the scope of this thesis, since they have a strong impact on very important parameters, such as on the duration of message transactions.

Next, the attributes of an Intermediate System (acting as a repeater), especially those influencing its timing behaviour, are defined. The start-relaying instant function, which characterises the instant when PhL PDUs start to be relayed (from the input port of the repeater to its output port) is characterised and its variation as a function of the length of Data Link Layer PDUs is analysed.

Finally, at the end of this chapter, there is a summary of the main issues that, as mainstream of this thesis, will be addressed in detail in Chapters 6, 7 and 8. These issues consist on: how to overcome the congestion problem in the Intermediate Systems (to which an innovative solution is devised in Chapter 6); how to appropriately set the Slot Time parameter and how to compute the worst-case duration of message transactions (to which detailed solutions are proposed in Chapter 7) and, finally, how to set the network parameters and how to perform the timing analysis concerning inter-cell mobility (issue addressed in detail in Chapter 8).

5.2. Models for the Communication Domains and End Systems

The objective of this section is to provide analytical models to characterise the Communication Domains and the End Systems, by defining a number of parameters (or attributes) for each component (object).

5.2.1. Sets of Components

The components of the Communication Network can be grouped into a number of sets:

Table 5.1: Sets of components in the Communication Network

Set	Description	Value
D	Set of Communication Domains in the Communication Network	$D = \{D^1,\ldots, D^{nd}\}$, where *nd* is the number of Communication Domains in the Communication Network.
ES	Set of End-Systems in the Communication Network	$ES = \{ES^1,\ldots, ES^{ne}\}$, where *ne* is the number of End Systems in the Communication Network
IS	Set of Intermediate Systems in the Communication Network	$IS = \{IS^1,\ldots, IS^{ni}\}$, where *ni* is the number of Intermediate Systems in the network
M	Set of Physical Media in the Communication Network	$M = \{M^1,\ldots, M^{nm}\}$, where *nm* is the number of Physical Media in the network

Figure 5.1 depicts a Communication Network with three Communication Domains (*nd* = 3), two Intermediate Systems (*ni* = 2), six End Systems (*ne* = 6) and three types of Physical Medium (*nm* = 3).

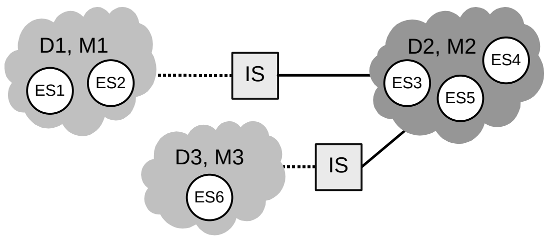


Figure 5.1: Sets of components in a specific layout of the Communication Network

5.2.2. Communication Domains

A Communication Domain (D^i) is defined as the set:

$$D^i = \left(D\_TYPE, M^l, IS(D^i), ES(D^i)\right) \tag{5.1}$$

with parameters as described in Table 5.2.

Table 5.2: Communication Domain parameters

Parameter	Description
D_TYPE	Represents the Communication Domain's type: $D_TYPE \in \{$WRD, MWRD, AWLD, SWLD$\}$ WRD – Wired Domain MWRD – Mobile Wired Domain AWLD – Ad-hoc Wireless Domain SWLD – Structured Wireless Domain
M^l	Physical medium of D^i $M^l \in M$
$IS(D^i)$	Function that returns the set of all ISs that are associated to D^i
$ES(D^i)$	Function that returns the set of all ESs that are associated to D^i

The Communication Domain model does not consider the three-dimensional "coverage map" of the Communication Domain, since it is not relevant within the scope of this thesis. The three-dimensional coverage map of a Wired Domain is the three-dimensional region covered by its wired medium (e.g. bus path). Similarly, the three-dimensional coverage map of a Wireless Domain is the three-dimensional region covered by its wireless medium (e.g. radio coverage region). Moreover, it would be necessary to define the absolute position of each Communication Domain in order to have complete knowledge of the three-dimensional coverage map of the Communication Network (to characterise the three-dimensional region covered by a set of interconnected Communication Domains).

5.2.3. End Systems

An End System (ES^j) is defined as the set:

$$ES^j = \left(ES\_TYPE, M^l, ES\_ROLE\right) \tag{5.2}$$

with parameters as described in Table 5.3.

Table 5.3: End System parameters

Parameter	Description
ES_TYPE	Represents the End System's type: $ES_TYPE \in \{$WRES, WLES, MWLES$\}$ WRES – Wired End System WLES – Wireless End System MWLES – Mobile Wireless End System
M^l	Physical medium of ES^j $M^l \in M$
ES_ROLE	Represents the End System's role $ES_ROLE \in \{$MASTER, SLAVE, MobM$\}$ MASTER – PROFIBUS master SLAVE – PROFIBUS slave MobM – Mobility master

5.2.4. Inter-cell mobility of End Systems

A Wireless End System is mobile (MWLES), if it has the capability to handoff between (structured) Radio Cells, i.e. of leaving a (Structured) Wireless Domain and joining another (Structured) Wireless Domain. A MWLES may be associated with no Wireless Domain, at a given moment in time. This happens if the MWLES is not in the range of any Wireless Domain that has the same physical medium as the MWLES. In this case, the MWLES is disconnected from the rest of the network.

For instance, in Figure 5.2 there is one Wired Domain (D^1) interconnecting two Structured Wireless Domains (D^2 and D^3) through two Structuring and Linking Intermediate Systems (SLIS). The Mobile Wireless End System (symbolised with an arrow, in Figure 5.2) can leave D^2 and join D^3, since D^2 and D^3 share the same physical medium (M^2).

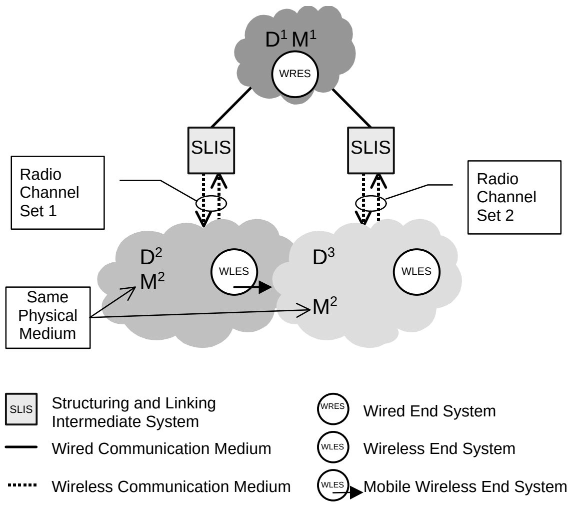


Figure 5.2: Mobile Wireless End System

5.2.5. Inter-cell mobility of Wired Domains

The mobility feature of a Wired Domain (MWRD) is always associated to the mobility feature of its Linking Intermediate System (LIS), and vice-versa. In a MWRD, the (one and only one) Linking Intermediate System must have the capability to handoff between (structured) radio cells, i.e. of leaving a (Structured) Wireless Domain and joining another (Structured) Wireless Domain. Therefore, it is denoted as Mobile Linking Intermediate System – MLIS.

Similarly to the case of a MWLES, a MLIS may be associated with no Wireless Domain, at a given moment in time. This happens if the MLIS is not in the range of any Wireless Domain that has the same physical medium as itself. In this case, the MWRD is disconnected from the rest of the network.

The Communication Network depicted in Figure 5.3 is similar to the one presented in Figure 5.2, with one additional Mobile Wired Domain (MWRD). This MWRD has a mobile Linking Intermediate System (MLIS) that is associated with Wireless Domain D^2. If the MWRD moves to the region of Wireless Domain D^3, the LIS will leave D^2 and join D^3 (D^2 and D^3 have the same physical medium – M^2).

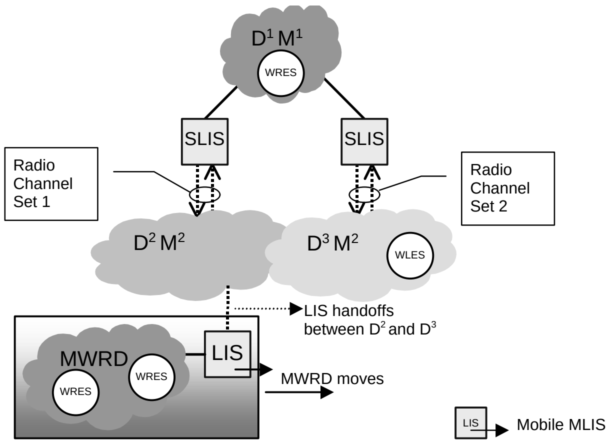


Figure 5.3: Mobile Wired Domain (MWRD)

5.3. Models for the Physical Media

In this section, a model for the Physical Media is proposed. Only after defining physical media parameters such as bit rate and PhL PDU (Physical Layer Protocol Data Unit) format it is possible to compute the duration of a PhL PDU or characterising the relaying behaviour of an Intermediate System.

5.3.1. Fundamentals of the model

A physical medium (M^l) is defined as a set:

$$M^l = \left(r^l, l_H^l, l_T^l, k^l\right) \tag{5.3}$$

with parameters as described in Table 5.4.

Table 5.4: Physical Media parameters

Parameter	Description	Units
r^l	Bit rate in physical medium l	Mbit/s
l^l_H	Overhead of the head per PhL PDU in physical medium l	bits
l^l_T	Overhead of the tail per PhL PDU in physical medium l	bits
k^l	Overhead per char for the PhL protocol of physical medium l	bits/char

A character (char) is defined as the smallest unit of information in the Data Link Layer (DLL). A DLL Protocol Data Unit (DLL PDU) is a set of chars delivered to the PhL for transmission. In order to proceed with this transmission, the PhL may have to introduce additional information ahead of the data field (head) and/or after the data field (tail). The head may include fields such as a preamble and a header, and its length is expressed by l^l_H. In turn, the tail may include fields such as a frame check sequence and an end delimiter, and its length is expressed by l^l_T.

The PhL may also have to add information to all or some DLL characters. A common example where additional information is included to all the DLL characters are the start, stop and parity bits. When encryption is under the responsibility of the PhL, it may be necessary to include additional (encryption) information in the data field of the PhL PDU. This additional information is usually added to groups of DLL characters (in blocks). Nevertheless, in the scope of this thesis, it will be considered an "average" overhead per DLL character - k^l, which must be computed for each PhL protocol.

5.3.2. Format of a PhL PDU

The generic format of a PhL PDU is as depicted in Figure 5.4. Note that the DLL PDU is embedded in the data field of the PhL PDU.

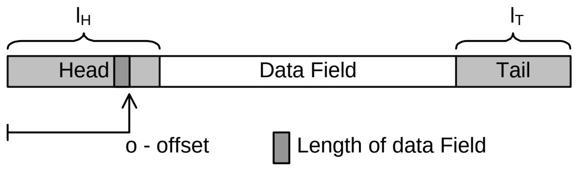


Figure 5.4: Generic format of a PhL PDU

Additionally, the offset o^l is defined as the total number of bits until knowing the length of the data field, for Physical Medium M^l. When the PhL PDU includes a head, the offset o^l is usually embedded inside. This offset parameter, formally characterised in Table 5.5, is a relevant parameter for the definition of the timing behaviour of the ISs (addressed in §5.4.3).

Table 5.5: Additional Physical Media parameter

Parameter	Description	Units
o^l	offset defining the total number of bits until knowing the length of the data field, in physical medium l	bits

If there is no PhL head, the length of data information must be found inside the data field, either in an implicit (e.g. through a unique identifier for each PDU length) or in an explicit (length of data field) way. The correct characterisation of this parameter is necessary for the definition of the Intermediate System model.

5.3.3. Duration of a PhL PDU

In order to compute the duration of a PhL PDU, two Data Link Layer parameters must be considered (Table 5.6). As such, they are not influenced by the Physical Media.

Table 5.6: DLL parameters for computing the duration of a PhL PDU

Parameter	Description	Units
L	Length of the DLL PDU	chars
d	Number of bits per char	bits/char

Taking into account the previously defined parameters and considering that for each Communication Domain there is one, and only one, associated Physical Medium, C^i is defined as the duration of a PhL PDU in domain D^i, and is given by:

$$C^i = \frac{l_H^i + L \cdot \left(d + k^i\right) + l_T^i}{r^i} \tag{5.4}$$

5.3.4. Assumptions and simplifications

Throughout this thesis, some simplifications to the Physical Media model are assumed. One concerns the propagation delays within the physical medium, which are considered negligible. Another one concerns the ESs, which are considered to have equal values for the minimum and maximum responders' turnaround times.

The propagation delay inside a Communication Domain can be neglected, since bit rates around 1-2 Mbit/s, bus segments smaller than 200 m and radio range shorter than 100 m are being considered. In copper media, assuming a propagation speed of 2×10^8 m/s, it takes 1 μs for a signal to propagate 200 m. The propagation delay for wireless communications is even smaller. Taking into account that, for a 1 Mbit/s bit rate a bit takes 1 μs to transmit and considering usual PhL PDU lengths of several tens/hundreds of bits, the propagation delay can be neglected. Additionally, the time elapsed between the reception of the last bit of a PhL request PDU and the transmission of the first bit of the correspondent PhL response/acknowledgement PDU – responder's turnaround time (t_{rt}) – is considered to be within the following bounded interval:

$$t_{rt} \in \left[t_{rt}^{\min}, t_{rt}^{\max}\right] \tag{5.5}$$

In the case of the PROFIBUS protocol, these limits correspond to the *min* T_{SDR} and *max* T_{SDR} DLL parameters described in §3.2.9, expressed in time units:

$$t_{rt}^{\min} = \frac{\min T_{SDR}}{r^l} \quad \text{and} \quad t_{rt}^{\max} = \frac{\max T_{SDR}}{r^l} \tag{5.6}$$

Throughout this thesis, it will be assumed that all the ESs in the Communication Network hold the same value for t^{min}_{rt} and the same value for t^{max}_{rt}.

5.4. Models for the Intermediate Systems

Within the proposed architecture, the Intermediate Systems (ISs) act as repeaters. For these particular components of the Communication Network, models will be significantly more complex, essentially due to their timing characterisation. The reasoning presented in this section is however crucial for the reminder of this thesis.

5.4.1. Fundamentals of the model

The proposed timing model enables the definition of a minimised latency repeater (cut-through behaviour), i.e., it permits to define a profile for a repeater that starts relaying PhL PDUs as early as possible. Additionally, a timing model for ISs operating in a "pure" store&forward fashion is also provided (which is a particular case of the generic model, where a PhL PDU must be completely received by the input port of the IS before being retransmitted to the output port. The ISs must also support some sort of encapsulation/decapsulation (due to different PhL PDU formats), be able to receive/transmit at different bit rates, and may also have to implement additional functionalities, such as encryption/decryption (security mechanisms at the PhL).

An Intermediate System (IS^k) is defined as the set:

$$IS^k = \left(IS\_TYPE, m^k, t_{rd}^k, T_{IDm}^k\right) \tag{5.7}$$

with the parameters as described in Table 5.7.

Table 5.7: Intermediate System parameters

Parameter	Description	Units
IS_TYPE	Represents the Intermediate System's type: $IS\_TYPE \in \{\text{SIS, LIS, MLIS, SLIS}\}$ SIS – Structuring Intermediate System LIS – Linking Intermediate System MLIS – Mobile Linking Intermediate System SLIS – Structuring & Linking Intermediate System	-
m^k	Set of the two physical media the IS interconnects. $m^k = \{M^{l1}, M^{l2}\}$	-
t^k_{rd}	Internal relaying delay	μs
T^k_{IDm}	Minimum idle (inactivity) time introduced by the IS between any two consecutive PhL PDUs.	bits

5.4.2. Assumptions and simplifications

The internal relaying delay – t^k_{rd} – is assumed to be independent both of the associated Communication Domains and of the DLL PDU that is being relayed (Figure 5.5). Moreover, for the remainder of this thesis, t^k_{rd} will be considered equal in every IS^k in the Communication Network. It is also assumed that the IS always introduces a minimum inactivity period – t_{IDm} (idle time) – between any consecutive PhL PDUs, according to the requirements of the PROFIBUS protocol (and of most fieldbus systems). Similarly, all the analysis carried out throughout this thesis assumes t^k_{IDm} equal in every IS^k in the Communication Network.

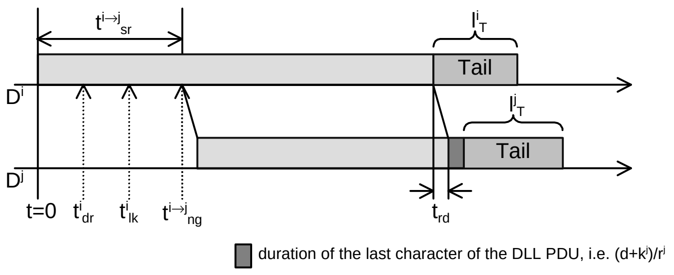


Figure 5.5: Timing behaviour of an IS

5.4.3. The start-relaying instant function

The start-relaying instant – $t^{i®j}_{sr}$ – is defined as the earliest time instant for start relaying a specific PhL PDU from Communication Domain D^i to Communication Domain D^j, counted since the beginning of the PhL PDU in Communication Domain D^i. The start-relaying instant for a specific IS depends on its envisaged behaviour – either store&forward or cut-through. For a cut-through IS, the following will be assumed:

- when relaying a PDU from D^i to D^j, it cannot start being relayed while the first char of the DLL PDU of D^i is not completely received by the IS;
- the PhL PDU cannot start being relayed while the length of the DLL PDU is not known (by the IS);
- when relaying a PDU from D^i to D^j, the instant for start relaying the PhL PDU must take into account that the IS cannot run out of bits to relay from D^i to D^j, i.e. the transmission of a PhL PDU in D_j must be continuous, without time gaps.

Taking these assumptions into account, which are illustrated in Figure 5.5, the start-relaying instant for a cut-through IS is defined as:

$$t^{i\to j}_{sr} = \max\left\{t^i_{dr}, t^i_{lk}, t^{i\to j}_{ng}\right\} \tag{5.8}$$

where:

- t^i_{dr}, the data ready instant, is the instant at which a predefined amount of DLL data has been received from D^i (ready to be relayed), counted since the beginning of the PhL PDU in D^i. For the cut-through behaviour, it is considered that it is the instant at which the first DLL char is completely received:

$$t^i_{dr} = \frac{l^i_H + k^i + d}{r^i} \tag{5.9}$$

- t^i_{lk}, the length known instant, is the instant at which the length of the DLL PDU in D^i is known, counted since the beginning of the PhL PDU in D^i. In this case, the offset value for the correspondent Physical Medium is used:

$$t^i_{lk} = \frac{o^i}{r^i} \tag{5.10}$$

- $t^{i®j}_{ng}$, the no gaps instant, is the earliest instant to start relaying the PhL PDU from D^i to D^j in a way that guarantees that the transmission in D^j is continuous. It may be computed by subtracting the duration of the PhL PDUs (neglecting the tail) in D^i and D^j and subtracting the duration of the last DLL PDU char in D^j (($d + k^j$) / r^j):

$$t_{ng}^{i\to j} = \frac{l_H^i}{r^i} - \frac{l_H^j}{r^j} + L\cdot\left(\frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right) - \frac{d+k^j}{r^j} \tag{5.11}$$

Consider the example depicted in Figure 5.5. The first time instant is data ready (t^i_{dr}), followed by the time instant when the length of the PDU is known (t^i_{lk}). The last instant (thus the maximum of the three) is the time instant that guarantees a continuous retransmission of the PhL PDU ($t^{i®j}_{ng}$). This situation usually happens when the duration of the PhL PDU in D^j is smaller than in D^i. Nevertheless, and for the general case, any of these time instants can be the maximum between them.

For the particular case of a store&forward behaviour, it is assumed that data is only ready to be relayed at the end of the PhL PDU, i.e. it is considered that $t^i_{dr} = C^i$, in the general model. This is depicted in Figure 5.6.

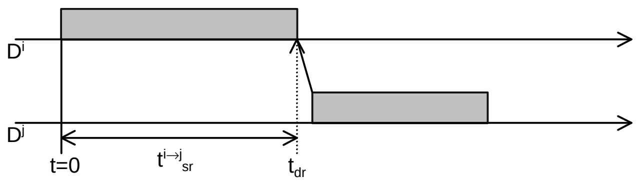


Figure 5.6: Timing behaviour of a store&forward IS

As a result, in Eq. 5.8, the data ready instant (t^i_{dr}) will assume the maximum value between (t^i_{dr}, t^i_{lk} and $t^{i®j}_{sr}$), and therefore the start-relaying instant will be (always) equal to the duration of the PDU:

$$t_{sr}^{i\to j} = C^i \tag{5.12}$$

5.4.4. Variation of the start-relaying instant, function of the length of DLL PDUs

An important characteristic of an IS is the non-linear variation of the start relaying instant (t_{sr}) as a function of the length of the DLL PDU (L). This behaviour is of paramount importance for the analysis carried out in Chapters 6 and 7. While in the particular case of a store&forward IS the start-relaying instant always grows with the length of a DLL PDU (L), in the general case this is not true, as it will be shown next.

For the general case, applying the derivative in order to L to $t^{i®j}_{sr}$ (Eq. 5.8), gives, for each of the three cases:

$$\begin{cases} \dfrac{\partial t^i_{dr}}{\partial L} = 0,\ \dfrac{\partial t^i_{lk}}{\partial L} = 0 \quad \text{(considering that } t^i_{dr} \text{ and } t^i_{lk} \text{ are constants)} \\ \dfrac{\partial t^{i\to j}_{ng}}{\partial L} = \partial\left(\dfrac{l^i_H}{r^i} - \dfrac{l^j_H}{r^j} + L\cdot\left(\dfrac{d+k^i}{r^i} - \dfrac{d+k^j}{r^j}\right) - \dfrac{d+k^j}{r^j}\right)\Big/\partial L = \dfrac{d+k^i}{r^i} - \dfrac{d+k^j}{r^j} \end{cases} \tag{5.13}$$

For the remainder of this thesis, and for the sake of simplicity, this derivative will be represented as (i.e., it will assume one of two possible values):

$$\frac{\partial t_{sr}^{i\mapsto j}}{\partial L} = \left\{ 0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j} \right\} \qquad (5.14)$$

This permits to conclude that:

$$\begin{cases} t_{sr}^{i\mapsto j} \text{ may increase with the increase in } L, \text{ if } \dfrac{d+k^i}{r^i} > \dfrac{d+k^j}{r^j} \\ t_{sr}^{i\mapsto j} \text{ may decrease with the increase in } L, \text{ if } \dfrac{d+k^i}{r^i} < \dfrac{d+k^j}{r^j} \end{cases} \qquad (5.15)$$

For the store&forward case, the following equation represents the derivative of $t^{i®j}_{sr}$ (Eq. 5.12) in order to L:

$$\frac{\partial t_{sr}^{i\mapsto j}}{\partial L} = \frac{\partial (C^i)}{\partial L} = \frac{d+k^i}{r^i} > 0 \qquad (5.16)$$

This means that the start-relaying instant for store&forward ISs always increases with the length of the DLL PDU (L), as already inferred from §5.4.3.

5.5. Models for the Communication Network and Message Streams

A number of message streams is considered to be associated to the End Systems. A Message Stream is the temporal sequence of message transactions, related, for instance, to the reading of a sensor. Message Streams fulfil the needs of communicating tasks in master ESs. In this section, Message Streams and also DLL and PhL parameters common to all components of the Communication Network will be characterised.

5.5.1. Message Streams

Each ES can have a set of Message Streams, where each of them is always associated to two ESs – the initiator and the responder (refer to Annex A, for the related OO model). In this sense, Table 5.1 must be complemented with the following set:

Table 5.8: Sets of components in the Communication Network

Set	Description	Value
S	Set of Message Streams in the Communication Network	$S = \{S^1,\ldots, S^{ns}\}$, where *ns* is the number of Message Streams in the Communication Network.

The computation of the PROFIBUS Slot Time parameter (T_{SL}) depends on the characteristics of every message stream in the Communication Network. Moreover, in order to carry out a worst-case message response time analysis, there is the need to compute the duration of a message transaction (addressed in detail in Chapter 7).

The relevant attributes of a Message Stream are (other, such as period and deadline, are not relevant for the analysis, in the scope of this Thesis):

$$S^s = \left(INITIATOR, RESPONDER, L_{req}, L_{resp}\right) \tag{5.17}$$

with parameters as defined in Table 5.9.

Table 5.9: Message Stream parameters

Parameter	Description	Units
INITIATOR	ES that is the initiator of the transaction *INITIATOR* Î $\{ES^1, \ldots, ES^{ne}\}$	-
RESPONDER	ES that is the responder of the transaction *RESPONDER* Î $\{ES^1, \ldots, ES^{ne}\}$	-
L_{req}	Length of the DLL request PDU	chars
L_{resp}	Length of the DLL response PDU	chars

5.5.2. Communication Network

There is a number of Communication Network parameters that must be defined, some related to the DLL and other concerning PhL behaviour. Although some of these parameters were already given sufficient intuition, some other are now introduced. Reasoning and intuition for their need will become clear later on, in the analysis provided in Chapters 6 and 7.

The model for the Communication Network is defined as:

$$N^n = \left(d, L_{req}^{\max}, L_{req}^{\min}, L_{resp}^{\max}, L_{resp}^{\min}, L_{token}, t_{rd}, T_{IDm}, t_{rt}^{\min}, t_{rt}^{\max}\right) \tag{5.18}$$

The parameters included in Eq. 5.18 are described in Table 5.10.

Table 5.10: Communication Network parameters

Parameter	Description	Units
d	Number of data bits per DLL char	bits
$L^{max}{}_{req}$	Maximum length of a DLL request PDU	chars
$L^{min}{}_{req}$	Minimum length of a DLL request PDU	chars
$L^{max}{}_{resp}$	Maximum length of a DLL response PDU	chars
$L^{min}{}_{resp}$	Maximum length of a DLL response PDU	chars
L_{token}	Length of the token PDU	chars
t_{rd}	Internal relaying delay of the IS	μs
T_{IDm}	Minimum inactivity (IDle) time introduced by the ES/IS between any two consecutive PhL PDUs.	bits
$t^{min}{}_{rt}$	Minimum responder's turnaround time	μs
$t^{max}{}_{rt}$	Maximum responder's turnaround time	μs

Note that parameter *d* was already defined in §5.3.3 (in order to compute the duration of a PhL PDU). If inter-cell mobility is supported by the Communication Network, five additional parameters must be defined (that was introduced in §4.5.2.

Table 5.11). These parameters are related to the mobility management mechanism that was introduced in §4.5.2.

Table 5.11: Additional Communication Network parameters

Parameter	Description	Units
L_{BT}	Length of the Beacon Trigger (BT) PDU	chars
nch	Number of radio channel sets	-
C_{beacon}	Duration of the beacon	µs
t_{bgap}	Time interval between beacons (beacon gap)	µs
t_{sw}	Switching delay between radio channels	µs

5.6. Putting the model into practice

This section presents an example (Figure 5.7), for which all the concepts and models previously outlined in this chapter will be bridged. The focus will be put on the components and layout of the Communication Network, with the purpose of giving the reader a less abstract approach. Details for timing parameters are not given at this stage.

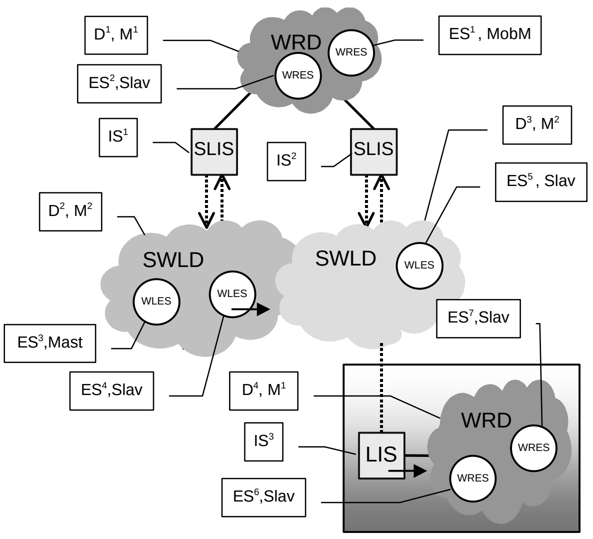


Figure 5.7: Example of a Communication Network

The components of the example Communication Network can be grouped into a number of sets, as depicted in Table 5.12.

Table 5.12: Sets of components of the Communication Network

$D=\{D^1,D^2,D^3,D^4\}$	$IS=\{IS^1,IS^2,IS^3\}$
$ES=\{ES^1,ES^2,ES^3,ES^4,ES^5,ES^6,ES^7\}$	$M=\{M^1,M^2\}$

The Physical Media models (M^1 and M^2) are not defined in this example. The Communication Domains are modelled (Eq. (5.1)) as given by Table 5.13.

Table 5.13: Model of the Communication Domains

$D^1 = (WRD, M^1, \{IS^1, IS^2\}, \{ES^1, ES^2\})$	$D^2 = (SWLD, M^2, \{IS^1\}, \{ES^3, ES^4\})$
$D^3 = (SWLD, M^2, \{IS^2, IS^3\}, \{ES^5\})$	$D^3 = (MWRD, M^1, \{IS^3\}, \{ES^6, ES^7\})$

where ESs are modelled (Eq. 5.2) as indicated in Table 5.14 and ISs (Eq. 5.7) as indicated in Table 5.15.

Table 5.14: Model of the End Systems

$ES^1 = (WRES, M^1, MobM)$	$ES^2 = (WRES, M^1, SLAVE)$	$ES^3 = (WLES, M^2, MASTER)$
$ES^4 = (MWLES, M^2, SLAVE)$	$ES^5 = (WLES, M^2, SLAVE)$	$ES^6 = (WRES, M^1, SLAVE)$
$ES^7 = (WRES, M^1, SLAVE)$		

Table 5.15: Model of the Intermediate Systems

$IS^1 = (SLIS, \{M^1, M^2\}, -, -)$	$IS^2 = (SLIS, \{M^1, M^2\}, -, -)$	$IS^3 = (MLIS, \{M^1, M^2\}, -, -)$

5.7. Main issues to be addressed in the architecture

The presented model for the architecture is the basis for addressing issues related to the timing behaviour of the Communication Network. These crucial issues will be introduced in the next three subsections, and solutions to these rather complex timing problems will be detailed in Chapters 6, 7 and 8.

5.7.1. Traffic adaptation by inserting additional idle time

Network interconnection often brings up the problem of network congestion. Generally, if for any time interval, the total sum of demands on a resource is more than its available capacity, the resource is said to be congested for that interval. In the case of computer networks, resources include buffer space and processing capacity in the Intermediate Systems (ISs) and link bandwidths (Jain, 1990). For instance, if during a short interval, the buffer space of an IS is smaller than the one required for the arriving traffic, frame loss may occur (dropped frames) and the IS is said to be congested.

It is also true that the congestion problems depend dramatically on the type of IS used in the interconnection. Particularly if ISs act as repeaters, traffic congestion may occur as a result of the heterogeneous characteristics of the interconnected physical media. The heterogeneity in bit rates and in PhL PDU formats in a broadcast (all messages received by all stations) network imposes the consideration of some kind of traffic adaptation scheme. This may solve the congestion problem while it also enables predictable and bounded message response times. While many congestion control and avoidance schemes have been proposed throughout the last two decades (Jacobson, 1988; Jain, 1990), they turn out to be unsuitable for field-level networks, and particularly to networking architectures such as the one addressed in this thesis.

The PROFIBUS MAC mechanism allows only one ES (in the Communication Network) to transmit at a given moment in time. However, the fact that different PhL PDU formats and bit rates exist on the different Physical Media may lead to cumulative pending messages in the IS, i.e. to traffic congestion.

The timing diagram depicted in Figure 5.8 illustrates a sequence of transactions between an initiator (who issues the request) – *I* – and a responder (who issues the response) – *R* – both in the same Communication Domain – D^i, and the resulting PDUs in the other Communication Domain – D^j. One Intermediate System interconnects the two Communication Domains and it is assumed that the PhL PDU duration in D^j is twice the PhL PDU duration in D^i and that $t^{i®j}_{sr}$ is constant (for the sake of simplicity). Note that since the idle time is defined as the duration of a predefined number of (idle) bits separating consecutive PDUs in the Communication Network, its duration is assumed to be different for the two Communication Domains.

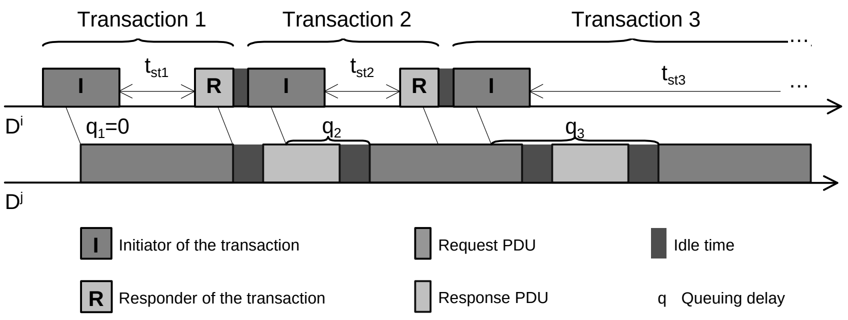


Figure 5.8: Increasing queuing delay in an intermediate system

Figure 5.8 illustrates an increasing queuing delay ($q_1 < q_2 < q_3$), caused by the different physical media, that will impact on the system turnaround time (t_{st}) for certain transactions. The *system turnaround time* – t_{st} – for a message transaction is the time elapsed since an Initiator ends transmitting a request PDU until it starts receiving the correspondent response PDU.

For instance, if the request correspondent to transaction 3 is addressed to a responder in Communication Domain D^j, the system turnaround time for this transaction (t_{st3}) will be affected by the cumulative queuing delay (q_3) in the Intermediate System. For the first two transactions, both the initiator and the responder(s) are in the same Communication Domain. Therefore, the system turnaround time for these transactions is equal to the responders' turnaround time ($t_{st1}= t_{st2}= t_{rt}$).

The queuing delay in the Intermediate System depends on a number of factors, namely on the number and duration of consecutive transactions where initiator and responder belong to D^i. Note that even a sequence of short length PDUs may lead to high queuing delays (thus to long worst-case message response times). For instance, a sequence of token passing between master ESs (in the same Communication Domain) that have no message requests to transmit may also cause traffic congestion.

Traffic congestion in Intermediate Systems can be avoided through the insertion of additional inactivity (idle) intervals before issuing message transactions. Obviously, the insertion of this additional idle time reduces the number of transactions per time unit when the responder is not in the same Communication Domain as the initiator. However, the advantage of avoiding traffic congestion is enormous. It leads to a better responsiveness to failure (when an error occurs, retransmissions are undertaken sooner) and to bounded and smaller worst-case message response times. Chapter 6 describes a novel solution for the traffic congestion problem, where every initiator inserts a predefined idle time either after receiving a response or token PDU (T_{ID1}) or after issuing an unacknowledged request PDU (T_{ID2}).

5.7.2. Computation the worst-case duration of message transactions and of the Slot Time parameter

All PROFIBUS master ESs have a Data Link Layer (DLL) parameter, the Slot Time - T_{SL} (refer to Section 3.2.10), which must be set (in all masters) before starting system's operation. This parameter defines the timeout before which a response/acknowledgement must arrive, and is also used for the token recovery mechanism.

For this purpose, it is necessary to compute the worst-case system turnaround time of message transactions (t_{st}), taking into account that a PDU sent by an initiator to a responder might have to be relayed by several Intermediate Systems.

Within this context of a hybrid wired/wireless communication network, T_{SL} assumes a particular importance. On one hand, T_{SL} must be set large enough to cope with the extra latencies introduced by the Intermediate Systems. On the other hand, T_{SL} must be set as small as possible such as the system responsiveness to failures does not decrease dramatically; that is, a master must detect a message/token loss or a station failure within a reasonably small time frame. Moreover, and in the context of a pre-run-time schedulability analysis of PROFIBUS messages, it becomes obvious that as T_{SL} is a time component of the worst-case duration of a message transaction (C), its value will impact on the evaluation of the worst-case message response time.

In order to fulfil these requirements, an optimal solution to set the PROFIBUS Slot Time parameter is proposed in Chapter 7. This solution is based on the computation of the worst-case system turnaround time of all message transactions and also permits to compute the worst-case duration of a message transaction (necessary for the worst-case message response time analysis).

5.7.3. Timing analysis of the mobility management mechanism

The basics of an innovative mechanism for supporting inter-cell mobility of mobile ESs and mobile Wired Domains were described in Chapter 4. In chapter 8, both the impact of supporting inter-cell mobility (using this mechanism) on the results obtained in Chapters 6 and 7 and the real-time behaviour of the proposed mechanism are analysed. A methodology to compute the mobility management duration, the appropriate idle time that the mobility master (MobM) must insert, and the number of beacons that each Structuring (& Linking) Intermediate System must issue is proposed.

Chapter 6

Inserting Idle Time to Adapt Heterogeneous Physical Media

One of the issues that emerges with the hybrid wired/wireless PROFIBUS-based communication network being specified is the adaptation of heterogeneous physical media. As explained before, the broadcast nature of the network leads to unpredictable and increasing queuing delays in the Intermediate Systems (ISs). This chapter proposes an innovative traffic adaptation scheme based on the adequate insertion of idle time between consecutive PDUs, by an appropriate setting of the PROFIBUS Idle Time parameters (inactivity times that master End Systems (ESs) respect before transmitting any request or token PDU).

6.1. Introduction

The heterogeneity in bit rates and PhL PDU (Physical Layer Protocol Data Unit) formats in a broadcast network requires the use of traffic adaptation mechanisms. In many kinds of LAN/WANs this problem is solved by the Intermediate Systems (ISs), usually acting as gateways, or as “intelligent” bridges controlling traffic generation in transmitting End Systems (ESs) or just discarding PDUs (Jacobson, 1988; Jain, 1990). However, in a broadcast fieldbus network (every transmitted PDU is received by every ES) with strict real-time and reliability requirements, a different approach must be followed. To our best knowledge, there is no previous relevant work focusing on a solution to this problem, and particularly in the framework of PROFIBUS networks.

This section proposes a solution where the responsibility of traffic adaptation is given to the master ESs, by inserting additional idle time between consecutive PDUs. It is also important to note that this analysis can generically be applied to any type of broadcast network, composed by heterogeneous transmission media and where ESs have the capability to insert idle time before issuing any request PDU (or token). Throughout the analysis, the network model described in Chapter 5 has been assumed.

It is important to stress that the inserted idle time guarantees that there is no increasing queuing in the ISs, but it does not avoid queuing delays in some ISs between initiator and responder of a transaction, or between a master ES and its successor when passing the token. This will only be demonstrated in Chapter 7, but we consider relevant to give some reasoning on that, at this moment, as described next.

1. It is proved (Section 7.3.2) that the maximum (worst-case) queuing delay affecting any request PDU occurs when this PDU is preceded by the maximum (worst-case) length PDU.

2. Therefore, the maximum queuing delay would occur for an infinite sequence of maximum length PDUs.
3. Nevertheless, it is proved (Annex C) that for $L_{req(l-1)} = L_{resp(l-1)} = L_{req(l)}$, there are no queuing delays (Q = 0).

Therefore, the inserted idle time guarantees no increasing queuing delay in the ISs. Moreover, by definition, the inserted idle time guarantees that there is no queuing delay in the first IS relaying the PDU from the initiator. From that IS on, there may exist queuing delay, but its worst-case (Q) can be computed (Sections 7.3.4 and 7.3.5).

6.2. Relevant time parameters

The PROFIBUS MAC mechanism allows only one ES to transmit at a given moment in time. However, the fact that different PhL PDU formats and bit rates exist on the different Physical Media may lead to cumulative pending messages in the IS.

Referring to the queuing delay problem introduced in Section 5.7.1, the most relevant time parameters and associated symbols involved are now going to be clarified. Consider the Communication Network illustrated in Figure 6.1 with an initiator (I) and a responder (R1) in the same Communication Domain (D^i) and another responder (R2) that belongs to the other Communication Domain – D^j.

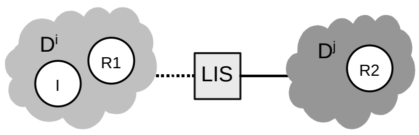


Figure 6.1: Example of a Communication Network

Figure 6.2 illustrates relevant time parameters for three consecutive transactions involving the initiator (I) in D^i and the two responders, one in D^i (R1) and the other in D^j (R2). These time parameters are crucial for the analytical models developed hereafter.

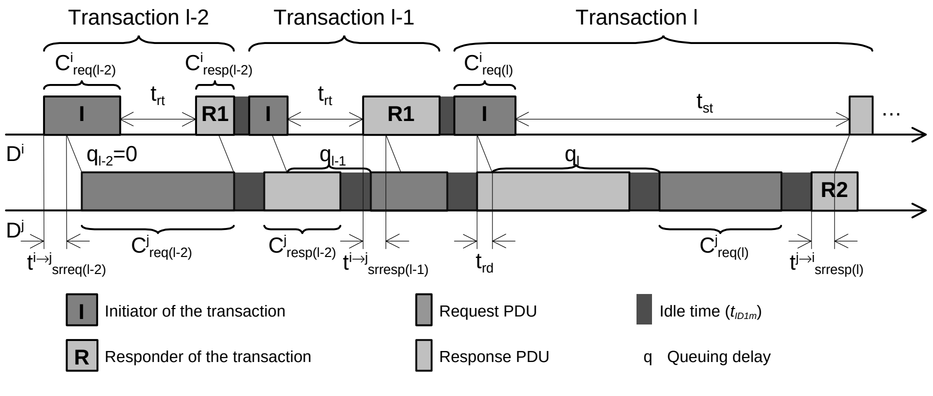


Figure 6.2: Variables relevant for the computation of the inserted idle time

In Figure 6.2, it is assumed that the duration (*C*) of the PhL PDUs in D^j is twice the duration in D^i and that the start relaying instant (t_{sr}) is constant (for the sake of simplicity). Note that since the idle time is defined as the duration of a predefined number of (idle) bits separating two consecutive PDUs in the Communication Network, its duration may be different for the two Communication Domains (refer to Chapter 3). Table 6.1 describes in detail all relevant time parameters.

Table 6.1: Notation and description of the time parameters

Notation	Description
$C^i_{req(l-2)}$	Duration of the request PDU of transaction (*l*-2) in Communication Domain *i*
$C^i_{resp(l-2)}$	Duration of the response PDU of transaction (*l*-2) in Communication Domain *i*
$C^j_{req(l-2)}$	Duration of the request PDU of transaction (*l*-2) in Communication Domain *j*
$C^j_{resp(l-2)}$	Duration of the response PDU of transaction (*l*-2) in Communication Domain *j*
$C^i_{req(l)}$	Duration of the request PDU of transaction *l* in Communication Domain *i*
$C^j_{req(l)}$	Duration of the request PDU of transaction *l* in Communication Domain *j*
t_{rt}	Responder's turnaround time for a message transaction
t_{st}	System turnaround time for a message transaction (t_{st} = t_{rt} for I and R in the same Communication Domain)
t_{rd}	Relaying delay of an IS, assumed to be equal for all ISs
$t^{i \text{®} j}_{srreq(l-2)}$	Start relaying instant of the request PDU of transaction (*l*-2) from Communication Domain *i* to Communication Domain *j*
$t^{i \text{®} j}_{srresp(l-1)}$	Start relaying instant of the response PDU of transaction (*l*-1) from Communication Domain *i* to Communication Domain *j*
$t^{j \text{®} i}_{srresp(l)}$	Start relaying instant of the response PDU of transaction *l* from Communication Domain *j* to Communication Domain *i*
q	Queuing delay for the corresponding transaction (*l*-2, *l*-1, *l*)

Every master ES in PROFIBUS holds two different Idle Time parameters – T_{ID1} and T_{ID2} (as already mentioned in Section 3.1, parameters denoted as 'T' are expressed in bit times while parameters denoted as 't' are expressed in seconds, throughout this thesis). A master ES always waits T_{ID1} after receiving a response/acknowledgement or a token PDU, before transmitting another PDU. It must also wait T_{ID2} after transmitting an unacknowledged request PDU, and before transmitting another PDU (request or token).

For a traditional wired network (just one Communication Domain), all master ESs may set their idle time parameters to the minimum default value (T_{ID1m} , T_{ID2m}), which is usually adequate to cope with bit synchronisation requirements.

The proposed traffic adaptation approach is based on the computation of the additional idle time that must be inserted by each master ES, in order to properly encompass the interconnection of heterogeneous physical media. These additional inserted idle times are represented by t_{ID1+} and t_{ID2+}. The remainder of this chapter will address the reasoning and the methodology for setting these two parameters in a generic Communication Network, as modelled in Chapters 4 and 5.

6.3. Computation of the inserted idle time after receiving a response

By inserting an appropriate inactivity (idle) time after receiving a response/acknowledgement PDU to an acknowledged request PDU, it is possible to guarantee that the next request PDU will experience no queuing delay in the first Intermediate System (IS).

Assume again Figure 6.2. If one considers that $q_{l-2} = 0$, it must be guaranteed that $q_{l-1} = 0$, $q_l = 0$, etc. In order to compute an appropriate value for the additional inserted idle time (t_{ID1+}), two time intervals are defined – $G^{i®j}_a$ and $G^{i®j}_b$ – along with the following reasoning: at the end of the $G^{i®j}_a$ interval, the request PDU is ready to be relayed by the IS, while at the end of the $G^{i®j}_b$ interval the IS is able to relay this request PDU. If $G^{i®j}_a < G^{i®j}_b$, then the request PDU will be affected by a queuing delay ($q > 0$).

More formally, $G^{i®j}_a$ is defined as the time elapsed from the beginning of transmission of the acknowledged request PDU of transaction *l-1* (*req*(*l-1*)) in Communication Domain *i* (D^i), until the moment when the request PDU of transaction *l* (*req*(*l*)) may start to be transmitted in Communication Domain *j* (D^j). Additionally, $G^{i®j}_b$ is defined as the time elapsed from the beginning of transmission of the acknowledged request PDU of transaction *l-1* (*req*(*l-1*)) in Communication Domain *i* (D^i), until the moment when the Intermediate System is able to start transmitting the request PDU of transaction *l* (*req*(*l*)) in Communication Domain *j* (D^j).

These two parameters ($G^{i®j}_a$, $G^{i®j}_b$) are illustrated in Figure 6.3. For the particular case, $G^{i®j}_a < G^{i®j}_b$, which results in a queuing delay ($q_l > 0$) for the request PDU of transaction *l*.

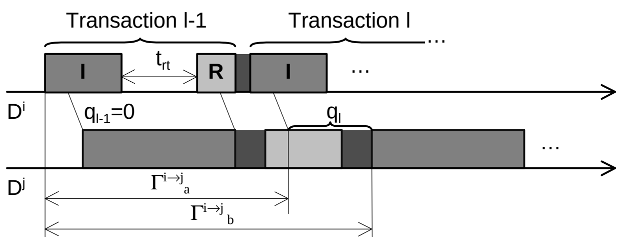


Figure 6.3: Queuing delay after an acknowledged transaction

To avoid such queuing delay in the IS, an additional inactivity time must be inserted by the initiator (I), in order to delay the request PDU of transaction *l*.

Therefore, it must be guarantee that:

$$\Gamma_a^{i\to j} \geq \Gamma_b^{i\to j},\quad \begin{cases} \forall\, i, j \in \{1,\ldots,nm\}\ \text{physical media} \\ \forall\, L_{req(l-1)}, L_{resp(l-1)} \in \ \text{DLL PDUs length from the message streams of I} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \tag{6.1}$$

where $L_{req(l-1)}$ is the length of the DLL request PDU for transaction (*l-1*), $L_{resp(l-1)}$ is the length of the DLL response PDU for transaction (*l-1*) and $L_{req(l)}$ is the length of the DLL request PDU (acknowledged or unacknowledged) for transaction *l* (or the length of the token PDU). These DLL PDU lengths correspond to the set of all DLL PDU lengths for the message streams of the master ES (I) whose inserted idle time is being computed.

To prevent a queuing delay in the IS ($G^{i®j}_{a}$ ³ $G^{i®j}_{b}$,), an additional idle time – $t^{i®j}_{ID1G+}$ must be inserted by each master ES (Figure 6.4a):

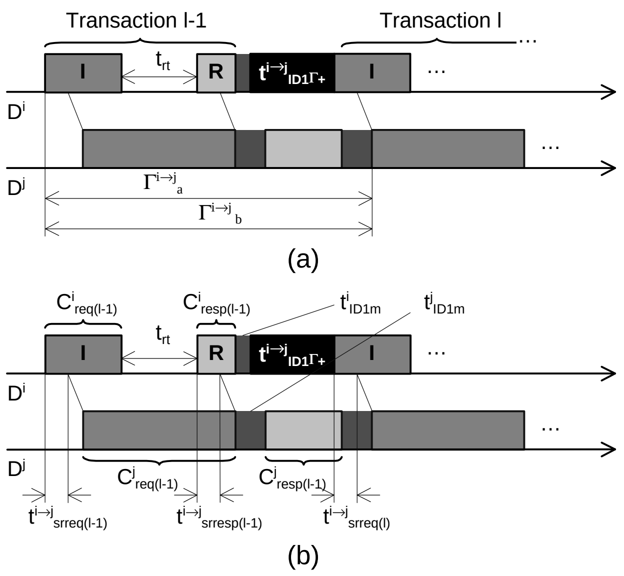


Figure 6.4: Inserting additional idle time after an acknowledged request

Using the parameters described in Table 6.1, which for clarification are represented in Figure 6.4b, $G^{i®j}_{a}$ may be computed as:

$$\Gamma_a^{i\to j} = C^i_{req(l-1)} + t_{rt} + C^i_{resp(l-1)} + t^i_{ID1m} + t^{i\to j}_{ID1\Gamma+} + t^{i\to j}_{srreq(l)} + t_{rd} \tag{6.2}$$

The computation of $G^{i®j}_{b}$ involves the maximum value between two parameters – $g^{i®j}_{a}$ and $g^{i®j}_{b}$ (Figure 6.5).

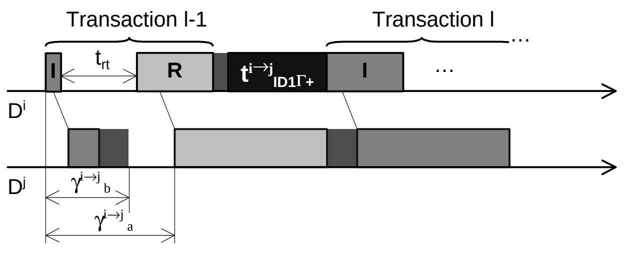


Figure 6.5: Components for the computation of $G^{i®j}_{b}$ (1)

The reasoning beyond these two time intervals is the following: at the end of the $g^{i®j}{}_{a}$ interval, the response PDU of transaction *l*-1 is ready to be relayed by the IS, while at the end of the $g^{i®j}{}_{b}$ interval the IS is able to relay this response PDU. In the example illustrated in Figure 6.5, $g^{i®j}{}_{a} > g^{i®j}{}_{b}$.

More formally, $g^{i®j}{}_{a}$ is defined as the time elapsed from the beginning of transmission of the acknowledged request PDU of transaction *l-1* (*req*(*l-1*)) in Communication Domain *i* (*D*^{i}), until the moment when the response PDU of transaction *l-1* (*resp*(*l-1*)) may start to be transmitted in Communication Domain *j* (*D*^{j}). Additionally, $g^{i®j}{}_{b}$ is defined as the time elapsed from the beginning of transmission of the acknowledged request PDU of transaction *l-1* (*req(l-1)*) in Communication Domain *i* (*D*^{i}), until the moment when the Intermediate System is able to start transmitting the response PDU of transaction *l-1* (*resp(l-1)*) in Communication Domain *j* (*D*^{j}).

Given these definitions, and again using the previously introduced (Table 6.1) time parameters, $g^{i®j}{}_{a}$ can be computed as follows:

$$g_{a}^{i\rightarrow j} = C^{i}_{req(l-1)} + t_{rt} + t^{i\rightarrow j}_{srresp(l-1)} + t_{rd} \qquad (6.3)$$

While $g^{i®j}{}_{b}$ is given by:

$$g_{b}^{i\rightarrow j} = t^{i\rightarrow j}_{srreq(l-1)} + t_{rd} + C^{j}_{req(l-1)} + t^{j}_{ID1m} \qquad (6.4)$$

Note that in the case depicted in Figure 6.4a, $g^{i®j}{}_{a} < g^{i®j}{}_{b}$. For clarification, that same scenario is reproduced in Figure 6.6, where both $g^{i®j}{}_{a}$ and $g^{i®j}{}_{b}$ are represented.

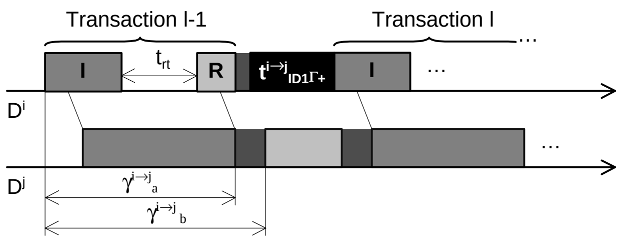


Figure 6.6: Components for the computation of $G^{i®j}{}_{b}$ (2)

Therefore, and taking into account the definition previously given $G^{i®j}{}_{b}$ and the ones given for $g^{i®j}{}_{a}$ and $g^{i®j}{}_{b}$, $G^{i®j}{}_{b}$ is analytically defined as:

$$\begin{aligned} &\Gamma_{b}^{i\rightarrow j} = \max\left\{g_{a}^{i\rightarrow j}, g_{b}^{i\rightarrow j}\right\} + C^{j}_{resp(l-1)} + t^{j}_{ID1m} = \\ &\max\left\{C^{i}_{req(l-1)} + t_{rt} + t^{i\rightarrow j}_{srresp(l-1)} + t_{rd}, t^{i\rightarrow j}_{srreq(l-1)} + t_{rd} + C^{j}_{req(l-1)} + t^{j}_{ID1m}\right\} + C^{j}_{resp(l-1)} + t^{j}_{ID1m} \end{aligned} \qquad (6.5)$$

Recalling that the inserted idle time after receiving a response PDU must be set in a way that guarantees that $G^{i®j}_{a}$ ³ $G^{i®j}_{b}$, then, using Eqs. (6.2) and (6.5) this inserted idle time is given by Eq. (6.6).

$$\begin{aligned} t_{ID1\Gamma+}^{i\to j} \geq\ & C_{req(l-1)}^{j} - C_{req(l-1)}^{i} + C_{resp(l-1)}^{j} - C_{resp(l-1)}^{i} + 2\cdot t_{ID1m}^{j} - t_{ID1m}^{i} - t_{rt} + \\ & + t_{srreq(l-1)}^{i\to j} - t_{srreq(l)}^{i\to j} + \\ & + \max\left\{ t_{srresp(l-1)}^{i\to j} - t_{srreq(l-1)}^{i\to j} + C_{req(l-1)}^{i} - C_{req(l-1)}^{j} + t_{rt} - t_{ID1m}^{j}, 0 \right\} \end{aligned} \tag{6.6}$$

Noting that, for a given master ES, the idle time must be set prior to run time and the same value will be used for all acknowledged transactions, $t^{i®j}_{ID1G+}$ must be set to the worst-case (maximum) scenario imposed by all the message streams of the master ES (I) under consideration. Therefore, t^{i}_{ID1G+} is defined as:

$$\begin{aligned} & t_{ID1\Gamma+}^{i} = \max\left\{ t_{ID1\Gamma+}^{i\to j} \right\} \\ & \begin{cases} \forall\, i, j \in \{1, \ldots, nm\} \text{ physical media} \\ \forall\, \mathrm{L}_{\mathrm{req}(l-1)}, \mathrm{L}_{\mathrm{resp}(l-1)} \in \text{DLL PDUs length from the message streams of I} \\ \forall\, \mathrm{L}_{\mathrm{req}(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \end{aligned} \tag{6.7}$$

Thus, further analysis is required on which set of $L_{req(l-1)}$, $L_{resp(l-1)}$, $L_{req(l)}$ leads to a maximum value for the inserted idle time. This can be approached by analysing the variation of the right side of inequality (6.6) as a function of the DLL PDU lengths involved ($L_{req(l-1)}$, $L_{resp(l-1)}$, $L_{req(l)}$). For that purpose, the derivatives of $t^{i®j}_{ID1G+}$ in order to $L_{req(l-1)}$, $L_{resp(l-1)}$ and $L_{req(l)}$ will be analysed. We will start with the derivative of the right side of inequality (6.6) in order to $L_{req(l-1)}$. Details and intermediate steps will be provided for this particular case. All the reasoning can then be borrowed to the derivatives in order to $L_{resp(l-1)}$ and $L_{req(l)}$. Therefore, in these cases intermediate steps are not provided.

Before applying the derivative to (6.6), since a *max* function exists, two separate forms will be used, for practical reasons. The first form assumes that the maximum between the two values under the *max* function will be 0 (case A). In this case, the form that the right side of inequality (6.6) assumes will be:

$$C_{req(l-1)}^{j} - C_{req(l-1)}^{i} + C_{resp(l-1)}^{j} - C_{resp(l-1)}^{i} + 2\cdot t_{ID1m}^{j} - t_{ID1m}^{i} - t_{rt} + t_{srreq(l-1)}^{i\to j} - t_{srreq(l)}^{i\to j}$$

In the case the maximum of the two values is greater than 0, then the right side of inequality (6.6) assumes the following form (case B):

$$\begin{aligned} & C_{req(l-1)}^{j} - C_{req(l-1)}^{i} + C_{resp(l-1)}^{j} - C_{resp(l-1)}^{i} + 2\cdot t_{ID1m}^{j} - t_{ID1m}^{i} - t_{rt} + \\ & + t_{srreq(l-1)}^{i\to j} - t_{srreq(l)}^{i\to j} + t_{srresp(l-1)}^{i\to j} - t_{srreq(l-1)}^{i\to j} + C_{req(l-1)}^{i} - C_{req(l-1)}^{j} + t_{rt} - t_{ID1m}^{j} \end{aligned}$$

which by aggregation of variables gives:

$$C_{resp(l-1)}^{j} - C_{resp(l-1)}^{i} + t_{ID1m}^{j} - t_{ID1m}^{i} - t_{srreq(l)}^{i\to j} + t_{srresp(l-1)}^{i\to j}$$

Therefore, the right side of inequality (6.6) can be re-written, that is, $t^{i®j}_{ID1G+}$ will be the maximum between the two following cases (A and B):

$$\begin{aligned}&\text{A: } C^{j}_{req(l-1)} - C^{i}_{req(l-1)} + C^{j}_{resp(l-1)} - C^{i}_{resp(l-1)} + 2\cdot t^{j}_{ID1m} - t^{i}_{ID1m} - t_{rt} + t^{i\rightarrow j}_{srreq(l-1)} - t^{i\rightarrow j}_{srreq(l)}\\ &\text{B: } C^{j}_{resp(l-1)} - C^{i}_{resp(l-1)} + t^{j}_{ID1m} - t^{i}_{ID1m} - t^{i\rightarrow j}_{srreq(l)} + t^{i\rightarrow j}_{srresp(l-1)}\end{aligned} \tag{6.8}$$

Also, before applying the derivatives to both forms of (6.8), it is important to note the following. In both cases, only the '*C*' and 't_{sr}' terms depend on the length of the DLL PDUs (*L*). For the '*C*' terms, the derivative of *C* in order to *L* will always be: $(d + k^i) / k^i$, for a PhL PDU in Communication Domain *i*, and *d*, k^i and r^i as defined in Tables 5.4 and 5.6. This result is borrowed from Eq. (5.16), for which reasoning is given in Chapter 5. For the case of the 't_{sr}' terms, the derivative of t_{sr} in order to *L* will always be the set: $\{0, (d + k^i) / r^i - (d + k^j) / r^j\}$, for a PDU being relayed from Communication Domain *i* to Communication Domain *j*. This result is borrowed from Eq. (5.14), for which reasoning is provided in Chapter 5.

Let us now workout the derivatives of both cases (A and B) in order to $L_{req(l-1)}$. For Case A, it develops as follows:

$$\begin{aligned}\frac{\partial\, t^{i\mapsto j}_{ID1\Gamma+}}{\partial L_{req(l-1)}} &= \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} + 0 - 0 + 2\times 0 - 0 - 0 + \left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\}\\ &= \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} + \left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\}\end{aligned}$$

Adding the first two terms to each of the two elements of the set, it results in:

$$\frac{\partial\, t^{i\mapsto j}_{ID1\Gamma+}}{\partial L_{req(l-1)}} = \left\{0, \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}\right\} \tag{6.9}$$

For case B of Eq. (6.8), the derivative will be 0 (no dependency on $L_{req(l-1)}$), i.e.:

$$\frac{\partial\, t^{i\mapsto j}_{ID1\Gamma+}}{\partial L_{req(l-1)}} = 0 \tag{6.10}$$

Taking into account Eqs. (6.9) and (6.10), one can conclude that:

$$\begin{cases} t^{i\mapsto j}_{ID1\Gamma+} \text{may increase with the increase of } L_{req(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i}\\ t^{i\mapsto j}_{ID1\Gamma+} \text{may decrease with the increase of } L_{req(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i}\end{cases} \tag{6.11}$$

Note that $(d + k^i) / r^i$ is the duration of a DLL character in D^i. From a less formal point of view, Eq. (6.11) states that if a character has a longer duration in D^j than in D^i, then the additional idle time that must be inserted ($t^{i®j}_{ID1G+}$) increases with the length of the DLL request PDU of the previous transaction ($L_{req(l-1)}$). Oppositely, if the duration of a character in D^i is greater than in D^j, then $t^{i\text{®}j}_{ID1G+}$ decreases with $L_{req(l-1)}$.

Now, the same reasoning will be developed for the derivatives of (6.8) in order to $L_{resp(l-1)}$ and $L_{req(l)}$.

Thus, applying the derivative in order to $L_{resp(l-1)}$ will give, for case A:

$$\frac{\partial t^{i\to j}_{ID1\Gamma+}}{\partial L_{resp(l-1)}} = \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} \tag{6.12}$$

While for case B it will be:

$$\frac{\partial t^{i\to j}_{ID1\Gamma+}}{\partial L_{resp(l-1)}} = \left\{ \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}, 0 \right\} \tag{6.13}$$

Considering the results expressed in Eqs. (6.12) and (6.13), one may conclude that:

$$\begin{cases} t^{i\to j}_{ID1\Gamma+} \text{may increase with the increase of } L_{resp(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ t^{i\to j}_{ID1\Gamma+} \text{may decrease with the increase of } L_{resp(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.14}$$

Finally, the derivative of (6.8) in order to $L_{req(l)}$, will be, for both cases of (6.8):

$$\frac{\partial t^{i\to j}_{ID1\Gamma+}}{\partial L_{req(l)}} = -\frac{\partial t^{i\to j}_{srreq(l)}}{\partial L_{req(l)}} = \left\{ 0, \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} \right\} \tag{6.15}$$

With which it may be concluded that:

$$\begin{cases} t^{i\to j}_{ID1\Gamma+} \text{may increase with the increase of } L_{req(l)}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ t^{i\to j}_{ID1\Gamma+} \text{may decrease with the increase of } L_{req(l)}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.16}$$

In summary, in order to evaluate the upper bound for the idle time that needs to be inserted by a given master ES, the following DLL PDU lengths must be chosen:

$$\begin{cases} L_{req(l-1)} = L^{\max}_{req}(S), L_{resp(l-1)} = L^{\max}_{resp}(S), L_{req(l)} = L^{\max}_{req}(S), & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ L_{req(l-1)} = L^{\min}_{req}(S), L_{resp(l-1)} = L^{\min}_{resp}(S), L_{req(l)} = L_{token}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.17}$$

with L^{max}_{req}, L^{max}_{resp} and L^{min}_{resp} as defined in Table 5.9 (Chapter 5), where S denotes the set of message streams for the master ES under consideration.

Obviously, in order to compute the value for $t^{i\text{®}j}_{ID1G+}$ for a given master ES, there is the need to know the characteristics of the message streams related to that master. Therefore, the length of the different DLL request and response/acknowledgement PDUs for every acknowledged request and the length of the different DLL request PDUs for every unacknowledged request must be known *a priori*. Note that, in PROFIBUS, the length of the token PDU (L_{token} = 3 chars) is always smaller than the minimum length of any request PDU (6 chars). That is the reason for using $L_{req(l)} = L_{token}$ (and not $L_{req(l)} = L^{min}_{req}(S)$) in the case where a minimum length is intended (the PDU of transaction *l* can either be a request or the token PDU).

It is also important to determine the value of the responder's turnaround time that maximises the inserted idle time. As $t^{i®j}_{ID1G+}$ increases with the decrease of t_{rt}, and considering that (from Eq. (5.5)) the responder's turnaround time can vary from t^{min}_{rt} to t^{max}_{rt}, t_{rt} must be set to t^{min}_{rt} in Eq. (6.6), in order to maximise the inserted idle time.

6.4. Computation of the inserted idle time after receiving the token

The condition expressed in Eq. (6.6) for the computation of $t^{i®j}_{IDG1+}$ is not sufficient for setting T_{ID1}, as a master ES must (also) insert T_{ID1} after the reception of a token PDU (refer to Section 3.2.9). In Figure 6.7, it is assumed that both the master ES that transmits the token and the master ES that receives it belong to Communication Domain *i* (D^i).

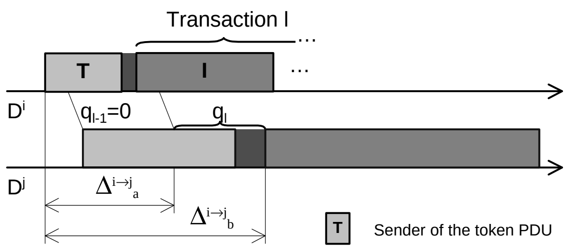


Figure 6.7: Queuing delay after receiving the token

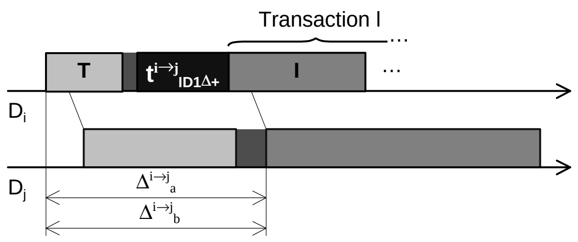
The interval $D^{i®j}_a$ is defined as the time elapsed from the beginning of transmission of the token PDU in Communication Domain *i* (D^i), until the moment when the request PDU of transaction *l* (*req*(*l*)) may start to be transmitted in Communication Domain *j* (D^j). Additionally, the interval $D^{i®j}_b$ is defined as the time elapsed from the beginning of transmission of the token PDU in Communication Domain *i* (D^i), until the moment when the Intermediate System is able to start transmitting the request PDU of transaction *l* (*req*(*l*)) in Communication Domain *j* (D^j).

In order to prevent queuing delay in the first IS, it must be guaranteed that:

$$\Delta_a^{i\to j} \geq \Delta_b^{i\to j}, \quad \begin{cases} \forall\, i, j \in \{1,\ldots,nm\} \text{ physical media} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \tag{6.18}$$

where $L_{req(l)}$ is the length of the DLL request PDU (may be an acknowledged request, an unacknowledged request or the token PDU) for transaction *l*.

It will be shown now how to compute the additional inactivity time – $t^{i®j}_{IDD1+}$ – that must be inserted after receiving the token PDU (Figure 6.8). The reasoning is similar to the one presented for the case of inserting additional idle time after a master ES receives a response PDU (i.e., analogue to the scenario presented in Figure 6.4).



Transaction l
T
$t^{i\rightarrow j}_{ID1\Delta+}$
l
…
D_i
D_j
$\Delta^{i\rightarrow j}_a$
$\Delta^{i\rightarrow j}_b$

Figure 6.8: Inserting additional idle time after receiving the token

The expression that permits to compute $D^{i®j}_a$ is:

$$\Delta_a^{i\rightarrow j} = C^i_{token} + t^i_{ID1m} + t^{i\rightarrow j}_{ID1+} + t^{i\rightarrow j}_{srreq(l)} + t_{rd} \tag{6.19}$$

Similarly, $D^{i®j}_b$ can be computed using the following expression:

$$\Delta_b^{i\rightarrow j} = t^{i\rightarrow j}_{srtoken} + t_{rd} + C^j_{token} + t^j_{ID1m} \tag{6.20}$$

The extra idle time that must be inserted after receiving the token can now be computed, imposing that the extra idle time guarantees that $D^{i®j}_a$ ³ $D^{i®j}_b$:

$$t^{i\rightarrow j}_{ID1\Delta+} \geq t^{i\rightarrow j}_{srtoken} - t^{i\rightarrow j}_{srreq(l)} + C^j_{token} - C^i_{token} + t^j_{ID1m} - t^i_{ID1m} \tag{6.21}$$

Similarly to Eq. (6.7), it is assumed that, for a given master station, the idle time must be set prior to run time and be valid for the worst-case scenario. Therefore, there is the need to find a maximum for $t^{i®j}_{ID1D+}$, i.e.:

$$t^i_{ID1\Delta+} = \max\left\{t^{i\rightarrow j}_{ID1\Delta+}\right\} \\ \begin{cases} \forall\, i, j \in \{1,\dots,nm\}\ \text{physical media} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of l or the token} \end{cases} \tag{6.22}$$

Let us determine values for $L_{req(l)}$ that lead to a maximum value for $t^{i®j}_{ID1D+}$. Note that contrarily to the case addressed in Section 6.3, here there is only need to analyse the characteristics of the request PDU following the token PDU; that is, $L_{req(l)}$. Thus, applying the derivative in order to $L_{req(l)}$ gives:

$$\frac{\partial t^{i\rightarrow j}_{ID1\Delta+}}{\partial L_{req(l)}} = -\frac{\partial t^{i\rightarrow j}_{srreq(l)}}{\partial L_{req(l)}} = -\left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\} = \left\{0, \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}\right\} \tag{6.23}$$

Therefore, it is possible to conclude that:

$$\begin{cases} t^{i\to j}_{ID1\Delta+} \text{ may increase with the increase of } L_{req(l)}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ t^{i\to j}_{ID1\Delta+} \text{ may decrease with the increase of } L_{req(l)}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.24}$$

Taking into account the results expressed in 6.24, in order to maximise $t^{i®j}_{ID1D+}$ for a given master ES, the following DLL PDU lengths must be chosen:

$$\begin{cases} L_{req(l)} = L^{\max}_{req}(S), & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ L_{req(l)} = L_{token}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.25}$$

For computing $t^{i®j}_{ID1D+}$ for a given master ES, there is the need to characterise the message streams related to that master. Therefore, the length of the different DLL PDUs for every acknowledged and unacknowledged request must be known *a priori*.

6.5. Setting the T_{ID1} parameter

The PROFIBUS T_{ID1} parameter is the idle time a master ES must insert after receiving a response PDU or after receiving the token PDU (refer to Section 3.2.9). Taking this into account, the inserted idle time $t^{i®j}_{ID1+}$ is defined as the maximum between $t^{i®j}_{ID1G+}$ (defined in Eq. 6.6) and $t^{i®j}_{ID1+}$ (defined in Eq. 6.21), i.e.:

$$t^{i\to j}_{ID1+} = \max\left\{t^{i\to j}_{ID1\Gamma+}, t^{i\to j}_{ID1\Delta+}\right\},$$
$$\begin{cases} \forall\, i, j \in \{1,\ldots,nm\} \text{ physical media} \\ \forall\, L_{req(l-1)}, L_{resp(l-1)} \in \text{DLL PDUs length from the message streams of I} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \tag{6.26}$$

In fact, in Eq. (6.26), $t^{i®j}_{ID1+}$ and $t^{i®j}_{ID1D+}$ are the minimum values that verify inequalities (6.6) and (6.21), respectively. Similarly, t^{i}_{ID1+} is defined as:

$$t^{i}_{ID1+} = \max\left\{t^{i}_{ID1\Gamma+}, t^{i}_{ID1\Delta+}\right\}, \quad \forall\, i \in \{1,\ldots,nm\} \text{ physical media} \tag{6.27}$$

Taking into account that the PROFIBUS protocol supports only one register for T_{ID1}, there is the need to aggregate both the "minimum" idle time T_{ID1m} (see Table 5.7) with the inserted idle time T_{ID1+} in one variable, for each master ES, i.e.:

$$T^{i}_{ID1} = T_{ID1m} + \left\lceil t^{i}_{ID1+} \cdot r^{i} \right\rceil \tag{6.28}$$

A simplified algorithm that returns the same idle time parameter values for all masters ESs in a given Physical Medium (therefore, in a per-Physical Medium basis) is presented in Annex B. For the sake of simplicity, instead of considering the particular set of message streams for each master station, a worst-case scenario where maximum and minimum PDU lengths for the (overall) Communication Network is considered.

6.6. Computation of the inserted idle time after sending an unacknowledged request

Adequate additional idle time must also be inserted after issuing an unacknowledged request. The PROFIBUS protocol uses the T_{ID2} parameter for this purpose (see Section 3.2.9). Figure 6.9 illustrates a scenario which will be used to provide a similar reasoning (as for T_{ID1}) for this additional idle time.

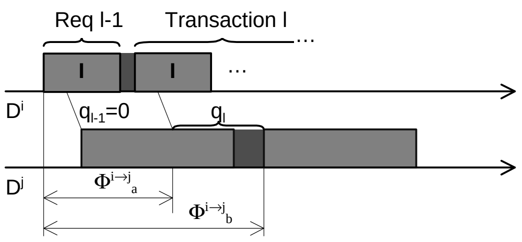


Figure 6.9: Queuing delay after unacknowledged request

In fact, the procedure to compute the inserted idle time after an unacknowledged request PDU is similar to the procedure adopted for computing the inserted idle time after receiving the token PDU (previous section).

$F^{i®j}_a$ is defined as the time elapsed from the beginning of transmission of the unacknowledged request PDU of transaction *l*-1 (*req*(*l-1*)) in Communication Domain *i* (D^i), until the moment when the request PDU of transaction *l* (*req*(*l*)) may start to be transmitted in Communication Domain *j* (D^j). Additionally, $F^{i®j}_b$ is defined as the time elapsed from the beginning of transmission of the unacknowledged request PDU of transaction *l*-1 (*req*(*l-1*)) in Communication Domain *i* (D^i), until the moment when the Intermediate System is able to start transmitting the request PDU of transaction *l* (*req*(*l*)) in Communication Domain *j* (D^j).

In order to prevent (increasing) queuing delays, it must be guaranteed that:

$$\Phi_a^{i\to j} \geq \Phi_b^{i\to j}, \qquad \begin{cases} \forall\, i, j \in \{1,\ldots,nm\}\ \text{physical media} \\ \forall\, L_{req(l-1)} \in \text{DLL PDUs length from the message streams of I} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \tag{6.29}$$

where $L_{req(l-1)}$ is the length of the DLL unacknowledged request PDU for transaction *l*-1 and $L_{req(l)}$ refers to an acknowledged request, to an unacknowledged request or to the token (assuming the value L_{token}, in this case) PDU for transaction *l*.

The additional inactivity time – $t^{i®j}_{ID2+}$ – must be set in a way that Eq. (6.29) is respected, with an example given in Figure 6.10.

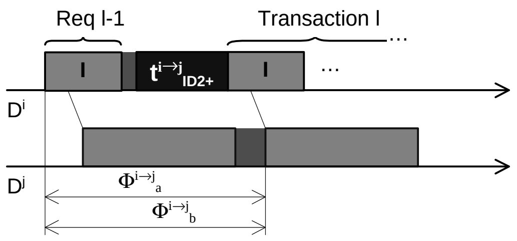


Figure 6.10: Inserting additional idle time after an unacknowledged request

The time interval parameter $F^{i®j}_a$ can be computed as:

$$\Phi_a^{i\to j} = C^i_{req(l-1)} + t^i_{ID2m} + t^{i\to j}_{ID2+} + t^{i\to j}_{srreq(l)} + t_{rd} \tag{6.30}$$

While the following expression corresponds to the computation of $F^{i®j}_b$:

$$\Phi_b^{i\to j} = t^{i\to j}_{srreq(l-1)} + t_{rd} + C^j_{req(l-1)} + t^j_{ID2m} \tag{6.31}$$

This means that (according to Eq. (6.29)):

$$t^{i\to j}_{ID2+} \geq t^{i\to j}_{srreq(l-1)} - t^{i\to j}_{srreq(l)} + C^j_{req(l-1)} - C^i_{req(l-1)} + t^j_{ID2m} - t^i_{ID2m} \tag{6.32}$$

Similarly to Eq. (6.7), the idle time must be set prior to run time, which implies finding a worst-case value for $t^{i®j}_{ID2+}$, i.e.:

$$t^i_{ID2+} = \max\left\{t^{i\to j}_{ID2+}\right\},$$
$$\begin{cases} \forall\, i, j \in \{1,...,nm\} \text{ physical media} \\ \forall\, L_{req(l-1)} \in \text{DLL PDUs length from the message streams of I} \\ \forall\, L_{req(l)} \in \text{DLL PDUs length from the message streams of I or the token} \end{cases} \tag{6.33}$$

Let us determine values for $L_{req(l-1)}$, $L_{req(l)}$ that lead to a maximum value for $t^{i®j}_{ID2}$. Applying the derivative to the right side of Eq. (6.32) in order to $L_{req(l-1)}$ develops as:

$$\frac{\partial\, t^{i\to j}_{ID2+}}{\partial L_{req(l-1)}} = \frac{\partial t^{i\to j}_{srreq(l-1)}}{\partial L_{req(l-1)}} + \frac{\partial C^j_{req(l-1)}}{\partial L_{req(l-1)}} - \frac{\partial C^i_{req(l-1)}}{\partial L_{req(l-1)}} =$$
$$= \left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\} + \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} = \left\{\frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}, 0\right\} \tag{6.34}$$

From Eq. (6.34), one can conclude that:

$$\begin{cases} t_{ID2+}^{i\mapsto j} \text{may increase with the increase of } L_{req(l-1)}, & \text{if } \frac{d+k^j}{r^j} > \frac{d+k^i}{r^i} \\ t_{ID2+}^{i\mapsto j} \text{may decrease with the increase of } L_{req(l-1)}, & \text{if } \frac{d+k^j}{r^j} < \frac{d+k^i}{r^i} \end{cases} \tag{6.35}$$

Now, applying the derivative in order to $L_{req(l)}$:

$$\frac{\partial t_{ID2+}^{i\mapsto j}}{\partial L_{req(l)}} = -\frac{\partial t_{srreq(l)}^{i\to j}}{\partial L_{req(l)}} = -\left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\} = \left\{0, \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}\right\} \tag{6.36}$$

which allows to conclude:

$$\begin{cases} t_{ID2+}^{i\mapsto j} \text{may increase with the increase of } L_{req(l)}, & \text{if } \frac{d+k^j}{r^j} > \frac{d+k^i}{r^i} \\ t_{ID2+}^{i\mapsto j} \text{may decrease with the increase of } L_{req(l)}, & \text{if } \frac{d+k^j}{r^j} < \frac{d+k^i}{r^i} \end{cases} \tag{6.37}$$

Taking into account the results expressed in (6.35) and (6.37), in order to maximise $t^{i®j}{}_{ID2}$ for a given master ES, the following DLL PDU lengths must be chosen:

$$\begin{cases} L_{req(l-1)} = L_{req}^{\max}(S), L_{req(l)} = L_{req}^{\max}(S), & \text{if } \frac{d+k^j}{r^j} > \frac{d+k^i}{r^i} \\ L_{req(l-1)} = L_{req}^{\min}(S), L_{req(l)} = L_{token}, & \text{if } \frac{d+k^j}{r^j} < \frac{d+k^i}{r^i} \end{cases} \tag{6.38}$$

To compute the value for $t^{i®j}{}_{ID2+}$ for a given master ES, there is the need to characterise the message streams related to that ES. Therefore, the length of the different DLL PDUs for every acknowledged and unacknowledged request must be known.

6.7. Setting the T_{ID2} parameter

Taking into account that the PROFIBUS protocol supports only one register for T_{ID2}, there is the need to aggregate the “minimum” idle time T_{ID2m} (see Table 5.9) with the inserted idle time T_{ID2+} in one variable, for each master ES, i.e.:

$$T_{ID2}^{i} = T_{ID2m} + \left\lceil t_{ID2+}^{i} \cdot r^{i} \right\rceil \tag{6.39}$$

The simplified algorithm presented in Annex B permits the computation of T_{ID2} in a per-domain basis.

6.8. Store&Forward Behaviour

For the case where Intermediate Systems have a store&forward behaviour, all the formulae for the computation of the inserted idle times apply. The only minor remark concerns the start relaying instant and its derivative in order to *L*, which for this case are expressed as in Eq. (5.12) and repeated here, for the sake of clarification:

$$t_{sr}^{i\to j} = C^i \quad \text{and} \quad \frac{\partial\, t_{sr}^{i\mapsto j}}{\partial L} = \frac{\partial\left(C^i\right)}{\partial L} = \frac{d+k^i}{r^i} \tag{6.40}$$

The change in the start relaying instant (and in its derivative in order to *L*) has an impact on the computation of the maximum value for the inserted idle time parameters, as explained next.

Taking $t^{i®j}{}_{sr} = C^i$ in Eq. (6.6) results in:

$$\begin{aligned} t_{ID1\Gamma+}^{i\to j} \geq\, & C_{req(l-1)}^{j} + C_{resp(l-1)}^{j} - C_{resp(l-1)}^{i} + 2\cdot t_{ID1m}^{j} - t_{ID1m}^{i} - t_{rt}^{\min} + \\ & - C_{req(l)}^{i} + \max\left\{C_{resp(l-1)}^{i} - C_{req(l-1)}^{j} + t_{rt}^{\min} - t_{ID1m}^{j}, 0\right\} \end{aligned} \tag{6.41}$$

The derivative of Eq. (6.41) in order to $L_{req(l-1)}$ is:

$$\frac{\partial\, t_{ID1\Gamma+}^{i\mapsto j}}{\partial L_{req(l-1)}} = \frac{d+k^j}{r^j} + \left\{-\frac{d+k^j}{r^j}, 0\right\} = \left\{0, \frac{d+k^j}{r^j}\right\} \tag{6.42}$$

This means that $t^{i®j}{}_{ID1G+}$ can only increase with $L_{req(l-1)}$.

The derivative of Eq. (6.41) in order to $L_{resp(l-1)}$ will be:

$$\frac{\partial\, t_{ID1\Gamma+}^{i\mapsto j}}{\partial L_{resp(l-1)}} = \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i} + \left\{\frac{d+k^j}{r^j}, 0\right\} = \left\{\frac{d+k^j}{r^j}, \frac{d+k^j}{r^j} - \frac{d+k^i}{r^i}\right\} \tag{6.43}$$

This result implies that:

$$\begin{cases} t_{ID1\Gamma+}^{i\mapsto j} \text{may increase with the increase of } L_{resp(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ t_{ID1\Gamma+}^{i\mapsto j} \text{may decrease with the increase of } L_{resp(l-1)}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.44}$$

Finally, the derivative of Eq. (6.41) in order to $L_{req(l)}$ is:

$$\frac{\partial\, t_{ID1\Gamma+}^{i\mapsto j}}{\partial L_{req(l)}} = -\frac{d+k^i}{r^i} \tag{6.45}$$

which means that $t^{i®j}{}_{ID1G+}$ can only decrease with the increase of $L_{req(l)}$.

Therefore, in order to define the appropriate value for $t^{i®j}_{ID1G+}$ for a given master ES, the following DLL PDU lengths must be chosen (worst-cases):

$$L_{req(l-1)} = L_{req}^{\max}, L_{req(l)} = L_{req}^{\max} \quad \text{and} \begin{cases} L_{resp(l-1)} = L_{resp}^{\max}, & \text{if } \dfrac{d+k^j}{r^j} > \dfrac{d+k^i}{r^i} \\ L_{resp(l-1)} = L_{resp}^{\min}, & \text{if } \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i} \end{cases} \tag{6.46}$$

Up to now, the analysis concerned the additional idle time after receiving a response PDU. For the case of the additional idle time after sending the token, the analysis is adapted as follows.

Taking $t^{i®j}_{sr} = C^i$ in Eq. (6.21) results in:

$$t_{ID1\Delta+}^{i\to j} \geq C_{token}^{j} - C_{req(l)}^{i} + t_{ID1m}^{j} - t_{ID1m}^{i} \tag{6.47}$$

For which the derivative in order to $L_{req(l)}$ will be:

$$\frac{\partial t_{ID1\Delta+}^{i\to j}}{\partial L_{req(l)}} = -\frac{d+k^i}{r^i} \tag{6.48}$$

This result means that $t^{i®j}_{ID1D+}$ can only decrease with the increase of $L_{req(l)}$. Therefore, in order to have a worst-case $t^{i®j}_{ID1D}$ for a given master ES, the following DLL PDU length must be chosen:

$$L_{req(l)} = L_{req}^{\min} \tag{6.49}$$

Finally, and concerning T_{ID2}, taking $t^{i®j}_{sr} = C^i$ in Eq. (6.32) results in:

$$t_{ID2+}^{i\to j} \geq C_{req(l-1)}^{j} - C_{req(l)}^{i} + t_{ID2m}^{j} - t_{ID2m}^{i} \tag{6.50}$$

Applying the derivative in order to $L_{req(l-1)}$:

$$\frac{\partial t_{ID2+}^{i\to j}}{\partial L_{req(l-1)}} = \frac{d+k^j}{r^j} \tag{6.51}$$

which means that $t^{i®j}_{ID2+}$ can only increase with the increase of $L_{req(l-1)}$. Applying the derivative in order to $L_{req(l)}$:

$$\frac{\partial t_{ID2+}^{i\to j}}{\partial L_{req(l)}} = -\frac{d+k^j}{r^j} \tag{6.52}$$

Taking into account the results expressed in (6.51) and (6.52), to have a worst-case $t^{i®j}_{ID2+}$ for a given master ES, the following DLL PDU lengths must be chosen:

$$\begin{cases} L_{req(l-1)} = L_{req}^{\max} \\ L_{req(l)} = L_{req}^{\min} \end{cases} \tag{6.53}$$

Obviously, the algorithm presented in Annex B can also be applied for the store&forward case, provided that the DLL PDU lengths presented in Eq. 6.17 are used (and the start relaying instant function is the one expressed in Eq. (6.40)). A slightly different approach for the computation of the inserted idle time was followed in (Alves *et al.*, 2001a), since exclusively the store&forward behaviour was considered, in that particular work.

6.9. Summary

In Chapter 5, it has been shown that if Communication Domains with different physical layers are interconnected using Intermediate Systems acting as repeaters, traffic congestion (in the Intermediate Systems) may occur. The problem originated by this heterogeneity in the physical layers is that message response times are affected by increasing delays due to queuing in the Intermediate Systems. Therefore, in order to have reduced and bounded message response times, a solution to this problem was proposed.

This novel solution this problem relies on an appropriate setting of the PROFIBUS Idle Time parameters, an inactivity time that must be respected by any master ES before issuing a request PDU (or passing the token). Obviously, the insertion of this additional idle time reduces the number of transactions per time unit, when the responder is not in the same domain as the initiator. Nevertheless, the advantage of avoiding traffic congestion is enormous. It leads to a better responsiveness to failure (when an error occurs, retransmissions are undertaken sooner) and to bounded and smaller worst-case message response times. Importantly, it was shown how to set the Idle Time parameters (T_{ID1} and T_{ID2}) for the cases where Intermediate Systems act as cut-through and store&forward repeaters.

The methodology presented in Sections 6.3-7 permits to set both idle time parameters in a per-station basis, taking into account all possible transactions (message streams) for that master ES. In this sense, each master ES in the Communication Network would have a unique pair (T_{ID1}, T_{ID2}) of idle time parameter values. For the sake of simplicity, a simplified algorithm that returns the same idle time parameter values for all master ESs in a given Physical Medium (i.e., in a per-Physical Medium basis) is presented in Annex B. Therefore, instead of considering the particular set of message streams for each master station, a worst-case scenario where maximum and minimum PDU lengths for the (overall) Communication Network is considered.

The elimination of traffic congestion by inserting additional idle time enables the computation of worst-case system turnaround times and message duration for any message transaction, permitting to set the PROFIBUS Slot Time parameter, as detailed in Chapter 7. It is also important to stress that the proposed mechanism does not impose any changes to the PROFIBUS protocol.

Chapter 7

Computing the Duration of Message Transactions and the Slot Time Parameter

A message transaction usually involves a request PDU followed by the correspondent response/acknowledgement PDU. Since in the proposed hybrid architecture, initiator and responder ESs may belong to different Communication Domains, request and response PDUs may have to be relayed by one or more ISs before reaching the destination. Therefore, a specific analysis is required to evaluate the additional latencies introduced by the broadcast nature of the network. This chapter presents a methodology to evaluate both the worst-case system turnaround time and the worst-case duration of any message transaction. It also presents a methodology on how to set the PROFIBUS Slot Time parameter.

7.1. Introduction

The message's response time in a PROFIBUS-based Communication Network is mainly dependent on the medium access delay (contention due to other messages in the queue and due to other stations holding the token) and on the duration of the message transaction. Such duration includes both the duration of the request/response PDUs and the system turnaround time associated with that transaction, that is, the time interval between the end of the request transmission and the beginning of the response reception.

While the medium access delay in fieldbus networks has been thoroughly studied in the last few years, the analysis of the system turnaround time has almost always been neglected. In a traditional wired network (just one Communication Domain), this may be reasonable. However, when considering the case of hybrid wired/wireless networks, there may exist several Communication Domains (and Intermediate Systems) between initiator and responder. Therefore, even considering that the Intermediate Systems (ISs) act as repeaters (broadcast network), system turnaround times can be several orders of magnitude higher than the duration of the request/response PDUs themselves.

The computation of the worst-case system turnaround time (for every possible message transaction in the network) permits also to compute one of the components of the PROFIBUS slot time parameter – T_{SL1} – in all master ESs. This parameter defines the timeout before which a response/acknowledgement must arrive (for every message transaction), and it is also used for the token recovery mechanism. The same reasoning, applied to the case where a master ES passes the token and waits for the next master station to transmit, permits to compute T_{SL2}. The Slot Time – T_{SL} – must be set to the maximum between T_{SL1} and T_{SL2}, prior to run-time. The reader is referred to Chapter 3 for further details on this parameter.

It is within this context that T_{SL} assumes a particular importance. On one hand, T_{SL} must be set large enough to cope with the extra latencies introduced by the ISs. On the other hand, T_{SL} must be set as small as possible, such as the system responsiveness to failures does not decrease dramatically; that is, a master must detect a message/token loss or a station failure within an acceptable time interval. Moreover, and in the context of a pre-run-time schedulability analysis of PROFIBUS messages (Tovar and Vasques, 1999a/b), it becomes obvious that as T_{SL} is a time component of the worst-case duration of a message transaction, its value will impact the evaluation of the worst-case message response time.

In order to fulfil such requirements, this chapter proposes methodologies in order to compute:

1. t_{st} – the worst-case system turnaround time for a message transaction;
2. T_{SL1} – one of the components of the PROFIBUS Slot Time parameter, based on the worst-case system turnaround time of all message transactions;
3. C_{ack} and C_{unk} – the duration of acknowledged and unacknowledged message transactions, respectively;
4. T_{SL2} – the second component of the PROFIBUS Slot Time parameter, based on the worst-case system turnaround time after a master passing the token to another master;
5. T_{SL} – the PROFIBUS slot time parameter, based on the maximum value between T_{SL1} and T_{SL2}.

In this chapter, it will be proved that the inserted idle time guarantees that there is no increasing queuing delays in the ISs. Nevertheless, there may occur queuing delays in some ISs (except the first) between initiator and responder of a transaction (or between a master ES and its successor, when passing the token). Consequently, the insertion of additional idle time enables the computation of the worst-case queuing delay – Q – affecting any request PDU. Such worst-case queuing delay will be a component of the worst-case system turnaround time for any message transaction ($t_{st} = Q + t_{stn}$), where t_{stn} is used to denote the system turnaround time assuming no queuing delay.

This chapter is structured as follows. Section 7.2 shows how to compute the system turnaround time of a message transaction assuming no queuing delay (t_{stn}) and analyses how the length of the request and response PDUs of that transaction impacts the t_{stn} value. Additionally, it also presents a methodology to compute the worst-case system turnaround time (t_{st}). Then, Section 7.3 shows how to compute the worst-case queuing delay (Q) for a message transaction, taking into account that the transaction may be preceded by an acknowledged (request/response) or unacknowledged (SDN) transaction. After, in Section 7.4, it is shown how to compute the duration of acknowledged (C_{ack}) and unacknowledged (C_{unk}) transactions.

Since the appropriate setting of the PROFIBUS Slot Time parameter is fundamental for the proper operation of the proposed Communication Network, Section 7.5 addresses the computation of its two components – T_{SL1} and T_{SL2}. Annex D presents two algorithms, one for the computation of T_{SL1} and of the worst-case duration of message transactions (C_{ack} and C_{unk}) and the other for the computation of T_{SL2}. Finally, the case where ISs behave as store&forward repeaters is addressed in Section 7.6.

7.2. Basics on the System Turnaround Time

Consider a Communication Network (Figure 7.1) with two End Systems (I and R), where every message transaction must be relayed by three Intermediate Systems (LISs).

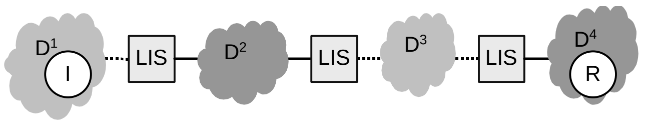


Figure 7.1: Example of a Communication Network

Figure 7.2 illustrates the time intervals for the computation of the system turnaround time of a transaction (*l*) between I and R. In the scenario, it is assumed that the request PDU of transaction *l* suffers no queuing delay in none of the IS:

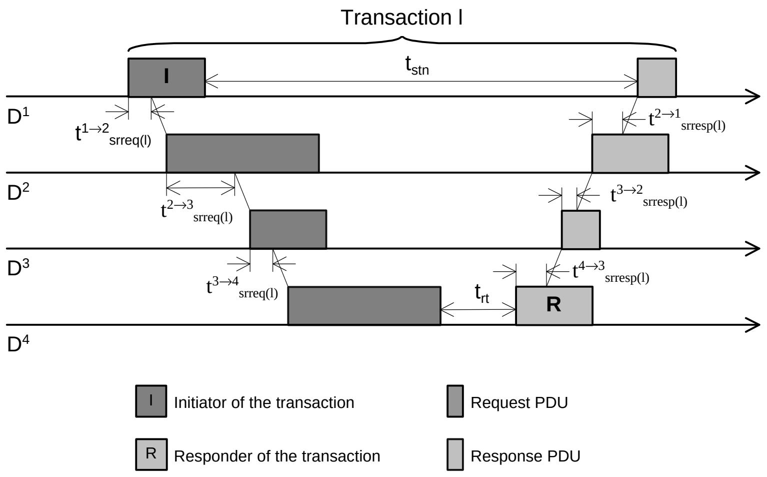


Figure 7.2: System turnaround time with no queuing delays

For this particular case, the system turnaround time is given by:

$$t^{1\to 4}_{stn(l)} = t^{1\to 2}_{srreq(l)} + t_{rd} + t^{2\to 3}_{srreq(l)} + t_{rd} + t^{3\to 4}_{srreq(l)} + t_{rd} + C^{4}_{req(l)} + t_{rt}$$
$$+ t^{4\to 3}_{srresp(l)} + t_{rd} + t^{3\to 2}_{srresp(l)} + t_{rd} + t^{2\to 1}_{srresp(l)} + t_{rd} - C^{1}_{req(l)}$$

Generically, the system turnaround time (assuming no queuing delays) of a transaction *l* between an initiator in Domain D^1 and a responder in Domain D^n is:

$$t^{1\to n}_{stn(l)} = \sum_{i=1}^{n-1}\left(t^{i\to i+1}_{srreq(l)} + t_{rd}\right) + C^{n}_{req(l)} + t_{rt} + \sum_{i=n}^{2}\left(t^{i\to i-1}_{srresp(l)} + t_{rd}\right) - C^{1}_{req(l)} \tag{7.1}$$

Considering that from Eq. (5.5), the responder's turnaround time can vary from t^{min}_{rt} to t^{max}_{rt}, t_{rt} must be set to t^{max}_{rt}, to consider the worst-case system turnaround time.

In order to set the Slot Time parameter in PROFIBUS and also to carry out a worst-case message response time analysis, it is necessary to determine the worst-case system turnaround time for all message transactions in the network. It must be stressed again that T_{SL} must be set with the same value in all master ESs of the Communication Network. To this purpose, the logical approach is to determine the worst-case system turnaround time for each master ES (taking into account all possible message streams for that ES) and then to choose the worst-case system turnaround time in the network (considering the overall set of masters).

Finally, the worst-case system turnaround time of a message transaction (t_{st}) is the sum of the worst-case total queuing delay affecting the request PDU (Q) and the system turnaround time assuming no queuing (t_{stn}):

$$t_{st} = Q + t_{stn} \tag{7.2}$$

In the following section, a methodology to compute the value of Q is proposed.

7.3. Computation of the worst-case queuing delay

When computing the system turnaround time for a message transaction, it is necessary to take into consideration that the request PDU of transaction *l* may be affected by queuing delays in the Intermediate Systems due to a previous PDU (response PDU or unacknowledged request PDU of transaction (*l*-1)).

7.3.1. Why there is no increasing queuing delays (bounded *Q*)

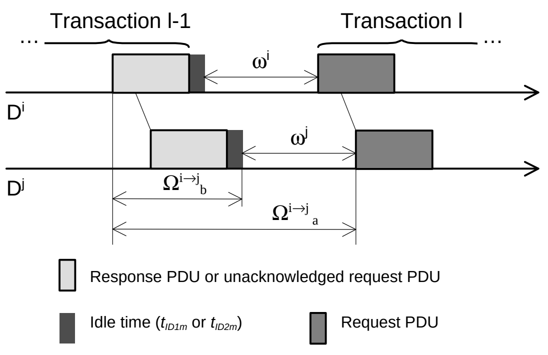
By definition, the inserted idle time guarantees that there is no queuing delay in the first IS to relay a PDU (request or token) from the initiator to the responder. From that IS on, this PDU may be affected by queuing delays, but the worst-case total queuing delay – Q – can be computed (as it will be shown in Sections 7.3.3 and 7.3.4). The following reasoning proves that the inserted idle time guarantees no increasing queuing delays:

1. in Section 7.3.2, it will be proved that the worst-case queuing delay affecting any (request) PDU occurs when it is preceded by a maximum length PDU;
2. therefore, the maximum queuing delay occurs for an infinite sequence of maximum (equal) length PDUs ($L^{max}_{req} = L^{max}_{resp} = L^{max}$);
3. nevertheless, it is proved in Annex C that if $L_{req(l-1)} = L_{resp(l-1)} = L_{req(l)}$, then there are no queuing delays affecting the request PDU of transaction *l* ($Q = 0$);
4. thus, the inserted idle time guarantees that there are no increasing queuing delays in the ISs.

7.3.2. How queuing delays grow with the length of the preceding PDU

It will be proved next that the queuing delay for a request PDU in transaction *l* grows with the increase in the length of the previous transaction's (*l*-1) PDU, which may be either a response PDU or an unacknowledged request PDU (Figure 7.3).

Assume that *i* is any Communication Domain but the one to which the station who issued the (*l*-1) PDU (as well as the station who issued the *l* request PDU) belongs to.



Transaction l-1
Transaction l
ω^i
D^i
ω^j
D^j
Ω^{i→j}_b
Ω^{i→j}_a
Response PDU or unacknowledged request PDU
Idle time (t_{ID1m} or t_{ID2m})
Request PDU

Figure 7.3: Queuing delay as a function of the preceding PDU

If the time span $w^j > 0$, then an infinitesimal increase in $L_{(l-1)}$ does not delay the request PDU of transaction *l* in D^i. However, if $w^j = 0$, an increase in $W^{i®j}_{b}$ may delay the request PDU of transaction *l* in D^i, causing an increase in the system turnaround time. Such influence can be analysed as follows.

For the case where $w^j > 0$, let us analyse how $L_{(l-1)}$ influences $W^{i®j}_{b}$:

$$\Omega_b^{i\to j} = t_{sr(l-1)}^{i\to j} + t_{rd} + C_{(l-1)}^{j} + t_{IDm}^{j} \tag{7.3}$$

Applying the derivative in order to $L_{(l-1)}$ results in:

$$\frac{\partial \Omega_b^{i\to j}}{\partial L_{(l-1)}} = \left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\} + \frac{d+k^j}{r^j} = \left\{\frac{d+k^j}{r^j}, \frac{d+k^i}{r^i}\right\} \tag{7.4}$$

Considering the first case of the set presented in Eq. (7.4):

$$\frac{\partial \Omega_b^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^j}{r^j} > 0 \tag{7.5}$$

For the second case:

$$\frac{\partial \Omega_b^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^i}{r^i} > 0 \tag{7.6}$$

Since both derivatives are always positive, $W^{i®j}_{b}$ always grows with the increase of $L_{(l-1)}$, which means that the system turnaround time of transaction *l* can only grow with the increase of $L_{(l-1)}$.

If $w^j = 0$, then an infinitesimal increase in $L_{(l-1)}$ will delay the request PDU of transaction *l* in Communication Domain *i*. Let us analyse how $L_{(l-1)}$ influences $W^{i®j}_{a}$. If it can be proved that $W^{i®j}_{a}$ increases with $L_{(l-1)}$, then the system turnaround time of transaction *l* also increases with $L_{(l-1)}$. $W^{i®j}_{a)}$ is given by the following expression:

$$\Omega_a^{i\to j} = \max\left\{C_{(l-1)}^{i} + t_{IDm}^{i} + t_{srreq(l)}^{i\to j} + t_{rd}, t_{sr(l-1)}^{i\to j} + t_{rd} + C_{(l-1)}^{j} + t_{IDm}^{j}\right\} \tag{7.7}$$

which can be separated into two cases. The first case is expressed in Eq. (7.8.)

$$\Omega_a^{i\to j} = C_{(l-1)}^i + t_{IDm}^i + t_{srreq(l)}^{i\to j} + t_{rd} \tag{7.8}$$

for which the derivative in order to $L_{(l-1)}$ results as follows:

$$\frac{\partial\Omega_a^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^i}{r^i} > 0 \tag{7.9}$$

For the second case:

$$\Omega_a^{i\to j} = t_{sr(l-1)}^{i\to j} + t_{rd} + C_{(l-1)}^j + t_{IDm}^j \tag{7.10}$$

where applying the derivative in order to $L_{(l-1)}$ results in:

$$\frac{\partial\Omega_a^{i\to j}}{\partial L_{(l-1)}} = \left\{0, \frac{d+k^i}{r^i} - \frac{d+k^j}{r^j}\right\} + \frac{d+k^j}{r^j} = \left\{\frac{d+k^i}{r^i}, \frac{d+k^j}{r^j}\right\} > 0 \tag{7.11}$$

As both derivatives are always positive, $W^{i®j}_a$ always grows with the increase of $L_{(l-1)}$, which means that the system turnaround time of transaction *l* can only grow with the increase of $L_{(l-1)}$. Since, for both cases ($w^j > 0$ and $w^j = 0$), $W^{i®j}$ increases with the increase of $L_{(l-1)}$, then it is proved that the system turnaround time of transaction *l* increases with the increase of $L_{(l-1)}$.

7.3.3. Why only the previous transaction must be considered

From Eq. (7.2), the worst-case system turnaround time of transaction *l* occurs when the request PDU of transaction *l* is subject to a maximum queuing delay (*Q*). It was proved (Section 7.3.2) that this maximum queuing delay occurs when the request PDU of transaction *l* is preceded by the longest PDU (response of an acknowledged transaction or unacknowledged request). Again, for this PDU (transaction (*l*-1)), the maximum queuing delay occurs when it is preceded by the longest PDU. This PDU may be either a request PDU if transaction (*l*-1) is acknowledged, or an unacknowledged request PDU (correspondent to transaction (*l*-2)), or the response of an acknowledged request (correspondent to transaction (*l*-2)). However, it has also been proved (Annex C) that for $L_{(l-1)} = L_{(l-2)}$, the request PDU of transaction (*l*-1) is not subject to queuing delays.

As a consequence, and from the temporal analysis point of view, this permits to "clear" the memory of the system (past transactions), i.e. for the computation of the worst-case total queuing delay (*Q*) affecting the request PDU of transaction *l*, only transactions (*l*-1) and *l* must be considered.

7.3.4. Computing queuing delays (transaction (*l*-1) is acknowledged)

Considering that transaction (*l*-1) is an acknowledged (request/response) transaction, its queuing delay in every IS between initiator and responder can be computed as follows. For this purpose, the definition of $G^{i®j}_a$ and $G^{i®j}_b$ (introduced in Section 6.3) will be extended to encompass more than one IS between initiator and responder.

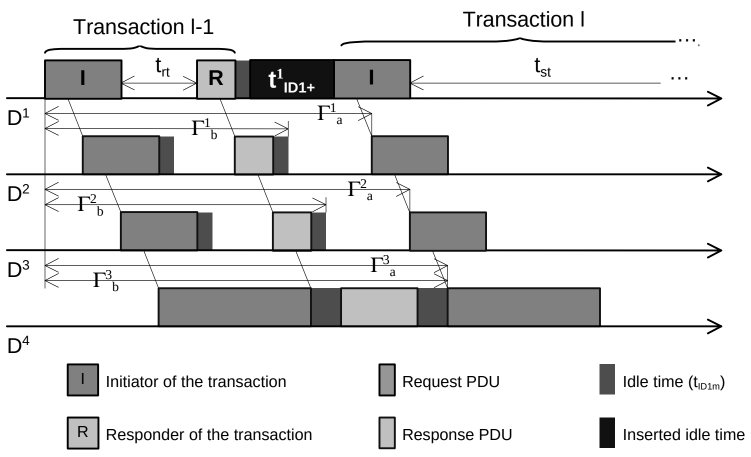


Figure 7.4: Extending the definition of $G^{i®j}_{a}$ and $G^{i®j}_{b}$ to G^{i}_{a} and G^{i}_{b}

G^{1}_{a} and G^{1}_{b}, which means considering the first IS in the path between the initiator and responder of transaction *l*, are given by the following formulae:

$$\begin{cases} \Gamma_a^1 = C^1_{req(l-1)} + t_{rt} + C^1_{resp(l-1)} + t^1_{ID1m} + t^1_{ID1+} + t^{1\to 2}_{srreq(l)} + t_{rd} \\ \Gamma_b^1 = \max\left\{C^1_{req(l-1)} + t_{rt} + t^{1\to 2}_{srresp(l-1)} + t_{rd}, t^{1\to 2}_{srreq(l-1)} + t_{rd} + C^2_{req(l-1)} + t^2_{ID1m}\right\} \\ \quad + C^2_{resp(l-1)} + t^2_{ID1m} \end{cases} \tag{7.12}$$

Considering the effect of the first two ISs in the path, then:

$$\begin{cases} \Gamma_a^2 = \max\left\{\Gamma_a^1, \Gamma_b^1\right\} + t^{2\to 3}_{srreq(l)} + t_{rd} \\ \Gamma_b^2 = \max\begin{Bmatrix} \Gamma_b^1 - C^2_{resp(l-1)} - t^2_{ID1m} + t^{2\to 3}_{srresp(l-1)} + t_{rd}, \\ t^{1\to 2}_{srreq(l-1)} + t_{rd} + t^{2\to 3}_{srreq(l-1)} + t_{rd} + C^3_{req(l-1)} + t^3_{ID1m} \end{Bmatrix} + C^3_{resp(l-1)} + t^3_{ID1m} \end{cases} \tag{7.13}$$

The generalisation for the case of a *i* number of consecutive ISs is computed as follows:

$$\begin{cases} \Gamma_a^i = \max\left\{\Gamma_a^{i-1}, \Gamma_b^{i-1}\right\} + t^{i\to i+1}_{srreq(l)} + t_{rd} \\ \Gamma_b^i = \max\begin{Bmatrix} \Gamma_b^{i-1} - C^i_{resp(l-1)} - t^i_{ID1m} + t^{i\to i+1}_{srresp(l-1)} + t_{rd}, \\ \sum_{j=1}^{i}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{i+1}_{req(l-1)} + t^{i+1}_{ID1m} \end{Bmatrix} + C^{i+1}_{resp(l-1)} + t^{i+1}_{ID1m} \end{cases} \tag{7.14}$$

where *i* represents the index in the set of Communication Domains belonging to the path between the initiator and responder of transaction *l*. Both G^{i}_{a} and G^{i}_{b} can be computed for any *i* *Î* *I* = {1,…, *ndp* - 1}, in which *ndp* is the number of Communication Domains in the referred path.

In Figure 7.4, it is assumed that the request PDU of transaction (*l*-1) suffers no queuing delay, in any of the IS. Since G^{i}_{a} ³ G^{i}_{b} for any Communication Domain *i* Î *I*, the request PDU of transaction *l* will not be subject to a queuing delay, in any of the IS. Now, consider another example (Figure 7.5) where $G^{i}_{b} > G^{i}_{a}$ for $i = 2$. The consequence is that the request PDU of transaction *l* is subject to a queuing delay in the second IS.

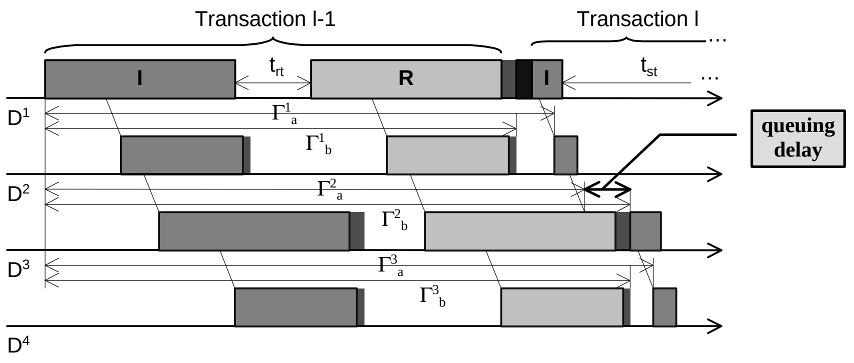


Figure 7.5: Example of queuing delay (from D^2 to D^3)

Obviously, this queuing delay contributes for the increase in the system turnaround time of transaction *l*. Generally, the queuing delay from Communication Domain *i* to Communication Domain (*i*+1) can be computed as follows:

$$q^{i}_{\Gamma} = \max\left\{0, \Gamma^{i}_{b} - \Gamma^{i}_{a}\right\} \tag{7.15}$$

In order to compute a worst-case value for the queuing delay, there is the need to determine the responder's turnaround time that leads to a maximum value for the queuing delay. It is easy to figure out that q^{i}_{G} increases with the decrease of t_{rt}. This is intuitive both from Figure 7.5, as well by a brief analysis of (7.12), by subtracting G^{i}_{a} to G^{i}_{b}. Taking into account that from Eq. (5.5), the responder's turnaround time can vary from t^{min}_{rt} to t^{max}_{rt}, t_{rt} must be set to t^{min}_{rt}, for use in Eq. (7.15), in order to get the worst-case queuing delay.

Therefore, the total queuing delay due to an acknowledged (*l*-1) transaction is:

$$Q_{\Gamma} = \sum_{j=1}^{ndp-1}\left(q^{j}_{\Gamma}\right) \tag{7.16}$$

where *ndp* is the number of Communication Domains in the path from the initiator to the responder.

7.3.5. Computing queuing delays (transaction (*l*-1) is unacknowledged)

Now, consider the case where the previous transaction (*l*-1) is an unacknowledged one. For that purpose, the definition of $F^{i®j}_{a}$ and $F^{i®j}_{b}$ (introduced in Section 6.6) will be extended to encompass more than one IS between initiator and responder (Figure 7.6).

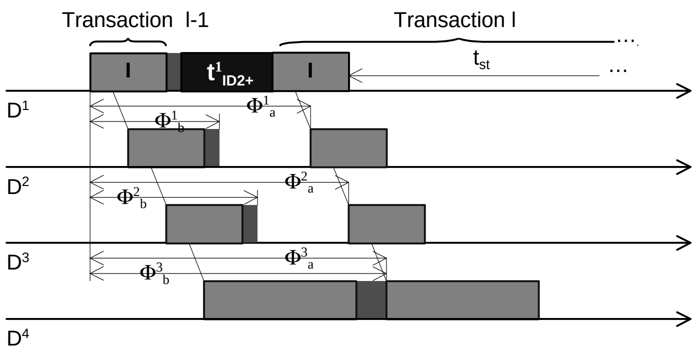


Figure 7.6: Extending the definition of $F^{i®j}_a$ and $F^{i®j}_b$ to F^i_a and F^i_b

F^1_a and F^1_b, which means considering the first IS in the path, are given by:

$$\begin{cases} \Phi^1_a = C^1_{req(l-1)} + t^1_{ID2m} + t^1_{ID2+} + t^{1\to 2}_{srreq(l)} + t_{rd} \\ \Phi^1_b = t^{1\to 2}_{srreq(l-1)} + t_{rd} + C^2_{req(l-1)} + t^2_{ID2m} \end{cases} \tag{7.17}$$

In the same way, F^2_a and F^2_b (that is, considering the first two ISs in the path) are:

$$\begin{cases} \Phi^2_a = \max\left\{\Phi^1_a, \Phi^1_b\right\} + t^{2\to 3}_{srreq(l)} + t_{rd} \\ \Phi^2_b = t^{1\to 2}_{srreq(l-1)} + t_{rd} + t^{2\to 3}_{srreq(l-1)} + t_{rd} + C^3_{req(l-1)} + t^3_{ID2m} \end{cases} \tag{7.18}$$

The generalisation for the case of a number *i* of consecutive ISs in the path is:

$$\begin{cases} \Phi^i_a = \max\left\{\Phi^{i-1}_a, \Phi^{i-1}_b\right\} + t^{i\to i+1}_{srreq(l)} + t_{rd} \\ \Phi^i_b = \sum_{j=1}^{i}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{i+1}_{req(l-1)} + t^{i+1}_{ID2m} \end{cases} \tag{7.19}$$

where *i* represents the index in the set of Communication Domains belonging to the path between the initiator and responder of transaction *l*. Both F^i_a and F^i_b can be computed for any $i \in I = \{1,\ldots, ndp - 1\}$.

For the example depicted in Figure 7.6, the request PDU of transaction (*l*-1) suffers no queuing delay, in any of the IS, as $F^i_a \geq F^i_b$ for any Communication Domain $i \in I$. However, similarly to the case where transaction (*l*-1) is acknowledged, F^i_b can be higher than F^i_a, leading to queuing delays that must be considered in the computation of the system turnaround time for transaction *l*. Therefore, the queuing delay from Communication Domain *i* to Communication Domain (*i*+1) can be computed as:

$$q^i_\Phi = \max\left\{0, \Phi^i_b - \Phi^i_a\right\} \tag{7.20}$$

Thus, the total queuing due to an unacknowledged (*l*-1) transaction is:

$$Q_\Phi = \sum_{j=1}^{ndp-1}\left(q^j_\Phi\right) \tag{7.21}$$

7.3.6. Computation of the worst-case total queuing delay (*Q*)

Finally, the worst-case total queuing delay *Q* can be computed as the maximum between the worst-case queuing delays imposed in case the precedent transactions are acknowledged (Q_G) or unacknowledged (Q_F), respectively, i.e.:

$$Q = \max\{Q_\Gamma, Q_\Phi\} \tag{7.22}$$

7.4. Computation of the duration of message transactions

In this section, a methodology for computing the duration of acknowledged and unacknowledged transactions will be provided.

7.4.1. Duration of request/response transactions (C_{ack})

The duration of an acknowledged transaction – C_{ack} (Figure 7.7); that is, a transaction constituted by request and response/acknowledgement PDUs, can be computed as:

$$C_{ack} = C^1_{Lreq} + t_{st} + C^1_{Lresp} + t^1_{ID1m} + t^1_{ID1+} \tag{7.23}$$

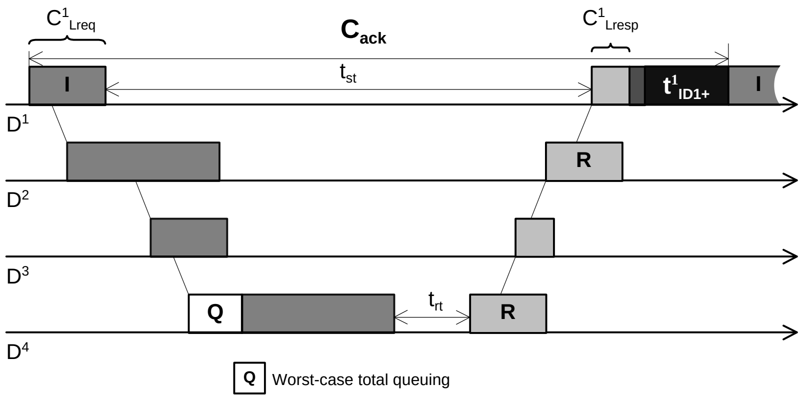


Figure 7.7: Duration of a request/response transaction (C_{ack})

7.4.2. Duration of unacknowledged transactions (C_{unk})

The duration of an unacknowledged transaction (C_{unk}), illustrated in Figure 7.8, does not depend on the path between the initiator and the responder, since there is no need to wait for any response/acknowledge for the initiator to issue another request (or token) PDU. Such duration can thus be evaluated as follows:

$$C_{unk} = C^1_{Lreq} + t^1_{ID2m} + t^1_{ID2+} \tag{7.24}$$

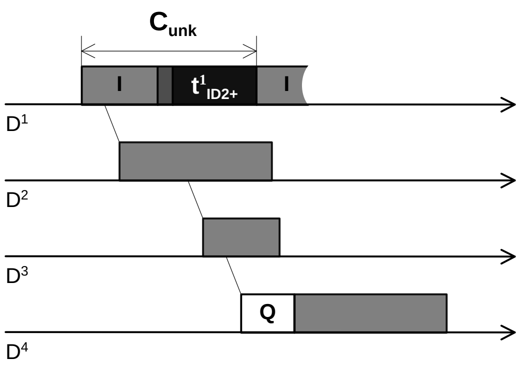


Figure 7.8: Duration of an unacknowledged transaction (C_{unk})

7.5. Computation of the Slot Time parameter

7.5.1. Computation of T_{SL}

Section 3.2.10 introduced the definitions of the Slot Time parameter and its two components – T_{SL1} and T_{SL2}. According to Eq. (3.6), all master ES in the Communication Network must set the Slot Time parameter to the same value, which is the maximum between t_{SL1} and t_{SL2}:

$$t_{SL} = \max\{t_{SL1}, t_{SL2}\} \tag{7.25}$$

In order to set the Slot Time parameter in every master ES, it is necessary to compute its value in bit times (T_{SL}) for every Communication Domain, as follows:

$$T_{SL}^i = \lceil t_{SL} \cdot r^i \rceil, \quad i \in D \tag{7.26}$$

Note that the computation of the Slot Time in bit times originates a (small) codification error that will turn the slot time value (in time units) different for Communication Domains with different bit rates.

The PROFIBUS standard states that the Slot Time parameter must be set to the same value in all master ES in the Communication Network. The justification for this relies on the procedure to reactivate a token loss, which depends on both T_{SL} and the ES address. The codification error mentioned before can be neglected, considering that its impact in the computation of the PROFIBUS Timeout parameter ($T_{TO} = 6 ´ T_{SL} + 2 ´ n ´ T_{SL}$, where n is the address of the master ES) is negligible.

7.5.2. Computation of T_{SL1}

The computation of t_{SL1} is quite straightforward, after having computed the worst-case system turnaround time for all message streams (*i* denotes the index of the message stream in the set *S*) in the Communication Network.

$$t_{SL1} = \max\{t_{st}(S[i])\} \tag{7.27}$$

The 11 bits and safety margin (T_{SM}) components defined in the PROFIBUS protocol (refer to Section 3.2.10) will be neglected. The reason for this is justified since these details are very much associated to the specifics of the RS-485 Physical Layer. When considering different types of physical layers, as in the addressed architecture, these details lose their meaning. Nevertheless, the sum of these adjustments is usually very small, comparing to the usual values of the worst-case system turnaround times.

The result from Eq. (7.27) should be used in Eq. (7.26), to compute T_{SL1} in bit times.

7.5.3. Computation of T_{SL2}

As described in Section 3.2.10, in order to compute T_{SL2} it is necessary to compute the worst-case system turnaround time after transmitting the token PDU – t_{st_token}. This is the maximum time the initiator waits after transmitting the last bit of the token PDU until it detects the first bit of a PDU (either a request or the token) transmitted by the ES that received the token. Figure 7.9 illustrates the time parameters that are used in the computation of t_{st_token}.

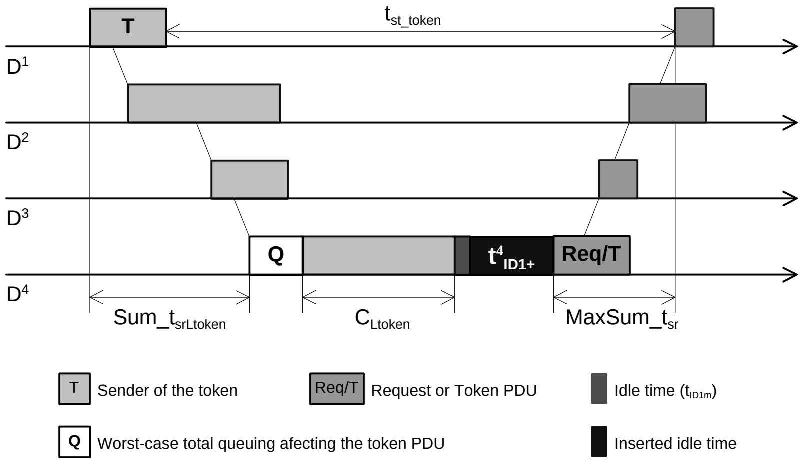


Figure 7.9: Illustration of t_{st_token}

The system turnaround time after transmitting the token PDU – t_{st_token} – can generically be computed as follows:

$$t_{st\_token} = Q + Sum\_t_{srLtoken} + C^{ndp}_{Ltoken} + t^{ndp}_{ID1m} + t^{ndp}_{ID1+} + MaxSum\_t_{sr} - C^{1}_{Ltoken} \tag{7.28}$$

where Q is the maximum total queuing affecting the token PDU, from the Communication Domain of the ES that transmitted the token to the Communication Domain of the ES that received the token.

The second component of Eq. (7.28) represents the time for relaying the token PDU along the previously referred path, and can be computed as follows:

$$Sum\_t_{srLtoken} = \sum_{j=1}^{ndp-1} \left(t_{srL_{token}}^{j \to j+1} + t_{rd} \right) \tag{7.29}$$

After receiving the token PDU, a master ES can issue a request PDU (belonging to the set of its message streams) or just pass the token to the next master ES. In order to have a maximum value for the system turnaround time, one must consider a worst-case scenario, i.e.:

$$Sum\_t_{sr}(Lreq) = \sum_{j=ndp}^{2} \left(t_{srL_{req}}^{j \to j-1} + t_{rd} \right) \tag{7.30}$$

This maximisation must consider all possible values for the length of the request DLL PDU (for all message streams of the master that received the token), including the length of the token DLL PDU, i.e. :

$$MaxSum\_t_{sr} = \max\left\{Sum\_t_{sr}(L_{req})\right\},$$
$$\forall \begin{cases} L_{req} = L_{token} \\ \text{or} \\ L_{req} \in \left\{L_{req}^{\min}, \ldots, L_{req}^{\max}\right\} \end{cases} \tag{7.31}$$

Finally, in order to compute T_{SL2} it is necessary to compute the worst-case system turnaround time after transmitting the token PDU for all master ESs ($M[i]$) in the Communication Network, i.e.:

$$t_{SL2} = \max\left\{t_{st\_token}\left(M[i]\right)\right\} \tag{7.32}$$

A simplification similar to the one assumed for the computation of t_{SL1} is assumed, i.e., the 11 bits and the safety margin (T_{SM}) components of T_{SL1} defined in the PROFIBUS protocol (refer to Section 3.2.10) are neglected.

T_{SL2} may then be computed using the result of Eq. (7.32) in Eq. (7.26).

7.6. Store&Forward Behaviour

For the case where Intermediate Systems have a store&forward behaviour, all the analysis, formulae, results and algorithms presented in this chapter are also valid. The only difference is that the start relaying instant is expressed as in Eq. (5.12) and its derivative in order to *L* as in Eq. (5.16), repeated here for convenience:

$$t_{sr}^{i\to j} = C^i \quad \text{and} \quad \frac{\partial\, t_{sr}^{i\to j}}{\partial L} = \frac{\partial\left(C^i\right)}{\partial L} = \frac{d+k^i}{r^i} \tag{7.33}$$

Therefore, the only results that must be reformulated are the ones presented in Eqs. (7.3) and (7.7). Eq. (7.3) is rewritten as follows:

$$\Omega_b^{i\to j} = C_{(l-1)}^i + t_{rd} + C_{(l-1)}^j + t_{IDm}^j \tag{7.34}$$

for which the derivative in order to $L_{(l-1)}$ results in:

$$\frac{\partial \Omega_b^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^i}{r^i} + \frac{d+k^j}{r^j} > 0 \tag{7.35}$$

Eq. 7.7 is rewritten as follows:

$$\Omega_a^{i\to j} = \max\left\{C_{(l-1)}^i + t_{IDm}^i + C_{req(l)}^i + t_{rd}, C_{(l-1)}^i + t_{rd} + C_{(l-1)}^j + t_{IDm}^j\right\} \tag{7.36}$$

which, similarly to Section 7.3.2, can be separated into two cases. In the first case:

$$\Omega_a^{i\to j} = C_{(l-1)}^i + t_{IDm}^i + C_{req(l)}^i + t_{rd} \tag{7.37}$$

for which the derivative in order to $L_{(l-1)}$ results in:

$$\frac{\partial \Omega_a^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^i}{r^i} > 0 \tag{7.38}$$

For the second case:

$$\Omega_a^{i\to j} = C_{(l-1)}^i + t_{rd} + C_{(l-1)}^j + t_{IDm}^j \tag{7.39}$$

where applying the derivative in order to $L_{(l-1)}$ results in:

$$\frac{\partial \Omega_a^{i\to j}}{\partial L_{(l-1)}} = \frac{d+k^i}{r^i} + \frac{d+k^j}{r^j} > 0 \tag{7.40}$$

Similarly to the case of the cut-through behaviour, both $W^{i®j}{}_a$ and $W^{i®j}{}_b$ increase with the increase of $L_{(l-1)}$. Therefore, also for this case it is proved that the system turnaround time of transaction *l* will also increase with the increase in $L_{(l-1)}$

Just to give a clearer picture of the computation of the system turnaround time for the case of the store&forward behaviour, an example is given in Figure 7.10. It is assumed that the request PDU of transaction *l* suffers no queuing delay in all the ISs.

Comparing to the case of Figure 7.2, the system turnaround time for the store&forward behaviour is considerably higher (note that PDU lengths are the same for both cases). For the store&forward case, Eq. (7.1) results in:

$$t_{stn(l)}^{1\to n} = \sum_{i=1}^{n-1}\left(C_{req(l)}^{i} + t_{rd}\right) + C_{req(l)}^{n} + t_{rt} + \sum_{i=n}^{2}\left(C_{resp(l)}^{i} + t_{rd}\right) - C_{req(l)}^{1} = \\ = \sum_{i=2}^{n}\left(C_{req(l)}^{i} + C_{resp(l)}^{i} + 2\cdot t_{rd}\right) + t_{rt} \qquad (7.41)$$

This result is equivalent to the one presented in (Alves et al., 2001b; Alves et al., 2002). Similarly to what was considered in Section 7.2, the responder's turnaround time that maximises the system turnaround time is $t_{rt} = t^{max}_{rt}$.

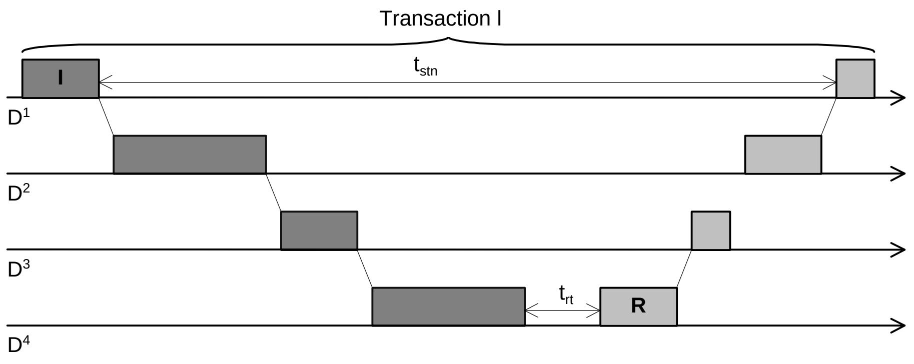


Figure 7.10: Example of system turnaround time for S&F (no queuing)

Oppositely to the cut-through behaviour, t_{stn} always grows with $L_{req(l)}$ and $L_{resp(l)}$, as it is going to be proved next. Applying the derivative in order to $L_{req(l)}$:

$$\frac{\partial t_{stn(l)}^{1\to n}}{\partial L_{req(l)}} = \sum_{i=2}^{n}\left(\frac{d+k^{i}}{r^{i}}\right) > 0 \qquad (7.42)$$

Now, applying the derivative in order to $L_{resp(l)}$:

$$\frac{\partial t_{stn(l)}^{1\to n}}{\partial L_{resp(l)}} = \sum_{i=2}^{n}\left(\frac{d+k^{i}}{r^{i}}\right) > 0 \qquad (7.43)$$

Since both derivatives are always positive, t_{stn} is maximum for $L_{req(l)} = L^{max}_{req}$ and $L_{resp(l)} = L^{max}_{resp}$.

Considering this result, the need for computing the worst-case total queuing (Q) could be avoided, by assuming that, for any message stream, $L_{req(l)} = L^{max}_{req}$. This is true since it is proved (in Annex C) that for $L_{req(l-1)} = L_{resp(l-1)} = L_{req(l)}$, the request PDU of transaction l is not affected by any queuing delays.

Is this case, the computation of the system turnaround time for transaction l would be reduced to:

$$t_{st(l)}^{1\rightarrow n} = \sum_{i=2}^{n} \left(C_{req}^{\max i} + C_{resp(l)}^{i} + 2 \cdot t_{rd} \right) + t_{rt} \tag{7.44}$$

It should be stressed that this is a slightly pessimistic result, since the effective length of the request PDU of transaction l is neglected, always using $L_{req(l)} = L^{max}_{req}$, (even if $L_{req(l)}$ is small). This approach was proposed in (Alves et al., 2001b; Alves et al., 2002).

7.7. Summary

Chapter 6 proposed a methodology to eliminate increasing queuing delays in the Intermediate Systems. Nevertheless, it does not avoid queuing delays beyond the first in the path between initiator and responder. By an appropriate setting of the PROFIBUS Idle Time parameters, it is possible to compute a worst-case (system) turnaround time and a worst-case duration for request/response transactions. This is of paramount importance for developing a worst-case message response time analysis and also for setting the PROFIBUS Slot Time parameter.

This chapter has shown how to compute the worst-case duration of acknowledged (request/response) and unacknowledged (SDN) message transactions (C_{ack} and C_{unk}, respectively), based on the knowledge that the worst-case queuing delay affecting any request PDU (Q) occurs when this PDU is preceded by PDUs with the maximum length.

By computing the worst-case system turnaround time (t_{st}) of all message transactions in the Communication Network, it is possible to set an appropriate value for the PROFIBUS Slot Time parameter (T_{SL}). This is a very important parameter in PROFIBUS networks, since it defines how long a master ES must wait for a response PDU before issuing a message retry (or abort that message transaction) or how long it should wait for any PDU after transmitting the token to another master ES. Both cut-through and store&forward behaviours were addressed.

Annex D presents two pseudo-code algorithms which implement the proposed formulae. One addresses the computation of the worst-case system turnaround time (t_{st}) and the worst-case duration of message transactions (C_{ack}), and of the first component of the PROFIBUS Slot Time parameter – T_{SL1}. The other algorithm permits the computation of the second component – T_{SL2}.

In this chapter, inter-cell mobility has still not been supported. The following chapter analyses how (inter-cell) mobility of MWLESs and MWRDs between Radio Cells impacts the results obtained in this chapter. Moreover, a timing analysis of the mobility management mechanism will be carried out.

Chapter 8

Timing Analysis Considering Inter-Cell Mobility

This chapter presents a timing analysis of the mobility management mechanism that was introduced in Chapter 4. As it will be shown, this mechanism has a time bounded duration. As mentioned in Chapter 4, the mechanism provides a seamless handoff for Mobile End Systems (masters and slaves) and for Mobile Linking Intermediate Systems (associated to Mobile Wired Domains). The impact of the inter-cell mobility in the Idle Time (Chapter 6) and Slot Time (Chapter 7) parameters will also be addressed.

8.1. Introduction

The basics of the Mobility Management mechanism adopted for the addressed hybrid wired/wireless communication network were already presented in Chapter 4. The main objective of this chapter is to find a methodology to set some important parameters for the Mobility Management mechanism.

In order for this mechanism to be compatible with the characteristics of PROFIBUS, after the Mobility Master (MobM) triggers the mobility management mechanism, it must insert an adequate idle time corresponding to the duration of the mobility management procedure, before issuing another transaction or passing the token. Since the Beacon Trigger (BT) PDU is a PROFIBUS SDN (unacknowledged request) PDU, the idle time to be inserted by the MobM after transmitting the BT PDU must be implemented by using T_{ID2}.

One of the main objectives of this chapter will therefore be the evaluation of the proper value to set T_{ID2} in the MobM. Obviously, this value is related to the worst-case duration of the mobility management procedure (which happens periodically). The duration of this mobility management procedure depends on the number of beacons that each Structuring Intermediate System (SIS) or Structuring & Linking Intermediate System (SLIS) must transmit, after having received (and relayed) the BT PDU. The number of beacons transmitted by each SIS/SLIS can be different, since they can receive the BT PDU at different instants (depending on the number of ISs and on the Physical Media in the path between the MobM and each SIS/SLIS). The way to compute the proper number of beacons to be transmitted by each SIS/SLIS is also a focus of this chapter.

This chapter starts by analysing the impact of supporting inter-cell mobility on previous results obtained in Chapters 6 and 7, before going into detail on the methodologies to compute the Mobility Management parameters.

8.2. Assumptions

The mobility management mechanism imposes that the MobM cannot be a mobile ES (MWLES) nor belong to a mobile WRD (MWRD), i.e. the relative physical position between the MobM and the SISs/SLISs cannot change. The reason for this is that the mobility management parameters (number of beacons of the SISs/SLISs and Idle Time inserted by the MobM) are computed prior to run-time. Therefore, there can be no changes in the path between the MobM and the SISs/SLISs.

The mobility management functionality can be under the responsibility of a dedicated master ES or can be integrated in a normal master ES. In the latter, the BT PDU coexists with the other message streams of that master ES. In the former, there is the guarantee that the MobM will pass the token to its successor, after issuing the BT PDU (and waiting a predefined idle time – T_{ID2}).

Additionally, all the Structured Wireless Domains (SWLD) are assumed to be associated to the same Physical Medium (i.e., same r, l_H, l_T, k, o parameter values). This simplifies the handoff procedure in the mobile ES and enables setting C_{beacon}, t_{bgap} and t_{sw} directly in time units (µs), in order to compute t_{ho}. Note that these parameters are defined in Table 5.10.

8.3. Impact of inter-cell mobility on previous results

8.3.1. Idle Time parameters

As it was explained in Chapter 6, the Idle Time parameters for a certain master ES are computed taking into account the different Physical Media existent in the Communication Network, the timing behaviour of the Intermediate Systems and on the set of Message Streams of that master ES. Neither the path between initiator and responder nor the path between a master and its successor are relevant. Therefore, there is no impact of considering inter-cell mobility on the computation of the Idle Time parameters. Nevertheless, it is important to note that the T_{ID2} parameter in the MobM must be computed in a different way (to the one presented in Chapter 6), as it will be described later on in this chapter.

8.3.2. Slot Time parameter

The computation of T_{SL1} depends on the worst-case system turnaround times of all message transactions in the Communication Network. For a particular message transaction, the fact that the initiator, the responder or both are mobile (or belong to a mobile WRD), and move between Radio Cells, can influence the worst-case system turnaround time for that message transaction. This is due to the change in the path, i.e. the ordered set of Communication Domains between initiator and responder depends on the location of both initiator and responder. Therefore, in order to compute the worst-case system turnaround time and the duration for a message transaction (and T_{SL1}), every possible path (due to inter-cell mobility) between initiator and responder must be considered.

Similarly, the computation of T_{SL2} is also influenced by mobility. The reason for this is that the path between a master and its successor can also vary, in case the master, its successor or both are mobile (or belong to a MWRD) and move between cells. Therefore, there is the need to compute the worst-case system turnaround time after passing the token, considering every possible path between the master and its successor, for every master in the Communication Network.

To clarify the impact of inter-cell mobility on the computation of the Slot Time parameter, consider the Communication Network layout given in Figure 8.1. Assume that there is one message transaction between WLES1 (initiator) and WLES2 (responder). In the scenario, the path between WLES1 and WLES2 is SWLD1 ($\rightarrow$ SLIS1) $\rightarrow$ WRD1 ($\rightarrow$ SLIS2) $\rightarrow$ SWLD2 ($\rightarrow$ LIS) $\rightarrow$ WRD2 ($\rightarrow$ SLIS3) $\rightarrow$ SWLD3. If WLES1 joins SWLD2, the path for the message stream changes to SWLD2 ($\rightarrow$ LIS) $\rightarrow$ WRD2 ($\rightarrow$ SLIS3) $\rightarrow$ SWLD3. If WLES1 moves to SWLD3, then the path is SWLD3 ($\rightarrow$ SLIS3) $\rightarrow$ SWLD3, i.e. involves only one IS, since initiator and responder are in the same Communication Domain (but every PDU must be relayed by the SLIS). Therefore, all these possible paths must be considered when computing T_{SL1}.

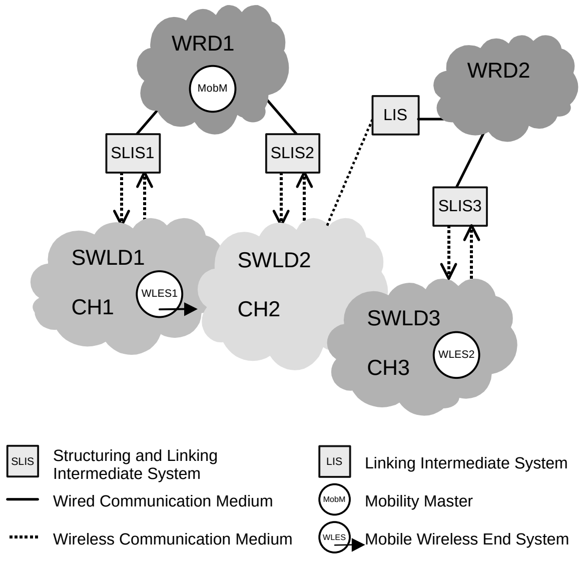

WRD1
MobM
WRD2
LIS
SLIS1
SLIS2
SLIS3
SWLD1
CH1
WLES1
SWLD2
CH2
SWLD3
CH3
WLES2
SLIS Structuring and Linking Intermediate System
LIS Linking Intermediate System
Wired Communication Medium
MobM Mobility Master
Wireless Communication Medium
WLES Mobile Wireless End System

Figure 8.1: Example of a Communication Network supporting inter-cell mobility

Similarly, in order to compute the second component of the Slot Time – T_{SL2} (which is related to the token-passing), the (inter-cell) mobility of master ESs must also be taken into account.

As an example, assume that two of the ESs in the example Communication Network depicted in Figure 8.1 are master stations – WLES1 and the MobM. The fact that WLES1 can join different SWLDs implies the consideration of the different paths from that master to its successor. In fact, the path of the token from WLES1 to the MobM is SWLD1 ($\rightarrow$ SLIS1) $\rightarrow$ WRD1 (if WLES1 is in SWLD1). If WLES1 moves to the coverage area of SWLD2, the path will be SWLD2 ($\rightarrow$ SLIS2) $\rightarrow$ WRD1.

8.3.3. Implementation approach

In order to compute the Slot Time parameter (T_{SL}) for a Communication Network supporting mobility, an implementation approach is proposed that introduces no changes to the algorithm for the computation of the Slot Time parameter presented in Annex E (no mobility support).

The implemented approach for the computation of T_{SL1} requires the definition of a different message stream for each possible path (between initiator and responder) of a particular message stream (as outlined in the input data file presented in Annex F). For instance, if a message stream can have 3 different paths between initiator and responder, 3 (distinct) message streams must be defined. Similarly, in the computation of T_{SL2}, it is necessary to define a different token passing stream for every possible path between a master and its successor. For instance, if 5 different paths between a master and its successor are possible, 5 (distinct) token passing streams must be defined.

Mobility is taken into account by considering these different paths in the input data, as exemplified in Chapter 9 and as reflected by the file presented in Annex F. This (text) file is used as input data for the program that computes all network parameters (e.g. T_{ID1}, T_{ID2}, t_{st} and C_{ack} for all message streams, T_{SL1}, T_{SL2} and the mobility management parameters).

8.4. Computation of the mobility management parameters

First, a preliminary value for the worst-case duration of the mobility management period (t'_{mob}) must be computed. This time span is defined as the time elapsed between the end of the transmission of the BT PDU (by the MobM) and the end of the handoff procedure in the MWLES/MLIS where this procedure ends last.

After having computed t'_{mob}, it is possible to determine the number of beacons each SIS/SLIS must transmit (n_b). Finally, the worst-case value for the mobility management duration (t_{mob}), from which the idle time (T_{ID2}) of the MobM can be computed, is readjusted considering the exact number of beacons sent by each SIS/SLIS. This readjustment results from the fact that SISs/SLISs must send an integer number of beacons that covers the worst-case period required for the MWESs/MLISs to perform channel assessment and switching.

8.4.1. Preliminary value for the mobility management duration (t'_{mob})

The MobM needs to insert an appropriate idle time before passing the token (in case the MobM is a dedicated master) or issuing another request PDU (in case the MobM is a normal master), in order to guarantee that the last mobile ES/LIS to receive the BT PDU has still enough time (sufficient number of beacons) to perform channel assessment and switching.

This value for the idle time is roughly the sum of the worst-case latency of the BT PDU (t_{bt}) and the worst-case duration of the handoff procedure (t_{ho}):

$$t'_{mob} = t_{bt} + t_{ho} \tag{8.1}$$

Figure 8.2 illustrates an example of the value of t'_{mob}.

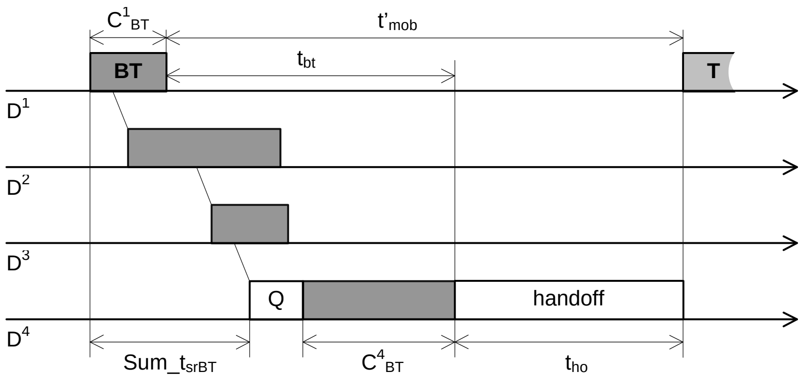


Figure 8.2: Example illustrating the value of t'_{mob}

The mobility management duration must be computed for every SWLD, in order to obtain the worst-case value, which is just a preliminary value for the worst-case mobility management duration. As mentioned before, this result must be readjusted depending on the (integer) number of beacons required to be sent by each SIS/SLIS. This will be explained later on in detail in Section 8.4.4.

8.4.2. Worst-case latency of the BT PDU (t_{bt})

The BT PDU takes a time interval t_{bt} to reach a specific SWLD, which is given by:

$$t_{bt} = Q + t_{btn} \tag{8.2}$$

where Q is the maximum queuing delay affecting the BT PDU, from the Communication Domain of the MobM to the SWLD being considered.

The second component (t_{btn}) represents the latency of the BT PDU along the previously referred path, without considering the queuing delay, and is given by:

$$\begin{cases} t_{btn} = Sum\_t_{srBT} + C_{BT}^{ndp} - C_{BT}^{1} \\ Sum\_t_{srBT} = \sum_{j=1}^{ndp-1}\left(t_{srBT}^{j\rightarrow j+1} + t_{rd}\right) \end{cases} \Rightarrow t_{btn} = \sum_{j=1}^{ndp-1}\left(t_{srBT}^{j\rightarrow j+1} + t_{rd}\right) + C_{BT}^{ndp} - C_{BT}^{1} \tag{8.3}$$

with definition of parameters as described in Table 8.1.

Table 8.1: Notation and description of relevant parameters

Notation	Description
ndp	Number of Communication Domains in the path between the MobM and an ES in the SWLD that is being considered
C^{ndp}_{BT}	Duration of the BT PDU in Communication Domain *ndp* (SWLD being considered)
C^{1}_{BT}	Duration of the BT PDU in Communication Domain 1 (location of the MobM)
Sum_t_{srBT}	Total latency of the BT PDU along the path from the MobM until the SWLD that is being considered
$t^{j®j+1}_{srBT}$	Start relaying instant of the BT PDU, from Communication Domain *j* to Communication Domain *j*+1
t_{rd}	Relaying delay of an IS, assumed equal for all ISs

In order to compute Q (the worst-case queuing delay for the BT PDU), two different cases must be considered, depending on the role of the MobM in the Communication Network:

1. the MobM is a normal master (with other message streams);
2. the MobM is exclusively dedicated to managing mobility.

The worst-case queuing delay for the Beacon Trigger PDU depends on these two cases. In the first case, the BT PDU may be preceded by any PDU from the message streams of that master. Therefore, it is assumed that the BT is preceded by maximum length PDUs either from a request/response transaction or an unacknowledged transaction. The computation of the queuing delay affecting the BT PDU is similar to the computation of the queuing delay for any request PDU (Chapter 7). This means computing Q_G and Q_F, considering $L_{req(l)} = L_{BT}$. Then, the worst-case total queuing delay – Q – in Eq. (8.2), is assumed to be the maximum between Q_G and Q_F.

Oppositely, if the MobM is exclusively dedicated to mobility management, the BT PDU is always preceded by the token PDU. In order to compute the queuing delay for this case, the definition of $D^{i®j}_{a}$ and $D^{i®j}_{b}$ (Section 6.4) will be extended to encompass the case of a token PDU being relayed through more than one IS, as illustrated in Figure 8.3.

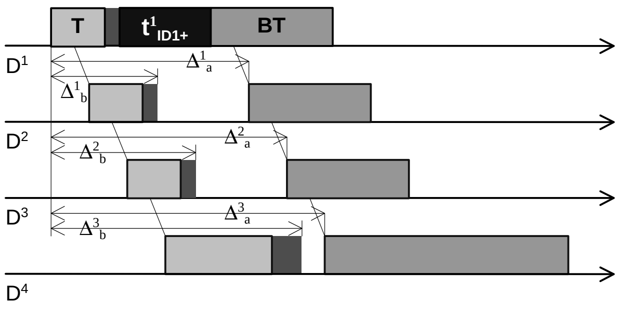


Figure 8.3: Extending the definition of $D^{i®j}_{a}$ and $D^{i®j}_{b}$ to D^{i}_{a} and D^{i}_{b}

The computation of the queuing delay is thus based on the computation of the related *D* intervals (Figure 8.3). Considering D^1_a and D^1_b, they may be computed as:

$$\begin{cases} \Delta^1_a = C^1_{token} + t^1_{ID1m} + t^1_{ID1+} + t^{1\to 2}_{srBT} + t_{rd} \\ \Delta^1_b = t^{1\to 2}_{srtoken} + t_{rd} + C^2_{token} + t^2_{ID1m} \end{cases} \quad (8.4)$$

and:

$$\begin{cases} \Delta^2_a = \max\{\Delta^1_a, \Delta^1_b\} + t^{2\to 3}_{srBT} + t_{rd} \\ \Delta^2_b = t^{1\to 2}_{srtoken} + t_{rd} + t^{2\to 3}_{srtoken} + t_{rd} + C^3_{token} + t^3_{ID1m} \end{cases} \quad (8.5)$$

Which generalising for any number of ISs in the path from the MobM to the SWLD:

$$\begin{cases} \Delta^i_a = \max\{\Delta^{i-1}_a, \Delta^{i-1}_b\} + t^{i\to i+1}_{srBT} + t_{rd} \\ \Delta^i_b = \sum_{j=1}^{i} \left(t^{j\to j+1}_{srtoken} + t_{rd}\right) + C^{i+1}_{token} + t^{i+1}_{ID1m} \end{cases}, \quad i \in I = \{1,\ldots,(ndp-1)\} \quad (8.6)$$

Consider that *i* represents the index in the set of Communication Domains belonging to the path between the MobM and an ES in the SWLD that is being considered. Both D^i_a and D^i_b can be computed for any $i \in I$ (1 to (*ndp* –1)), in which *ndp* is the number of Communication Domains in the referred path.

Similarly to the analysis of Chapter 7, the queuing delay introduced from Communication Domain *i* to Communication Domain (*i*+1) is given by:

$$q^i_\Delta = \max\{0, \Delta^i_b - \Delta^i_a\} \quad (8.7)$$

Consequently, the total queuing delay due to the token PDU can be computed as:

$$Q_\Delta = \sum_{i=1}^{ndp-1} q^i_\Delta \quad (8.8)$$

Therefore, for the second case of the set in Eq. (8.7), considering that the worst-case total queuing delay – *Q* – in Eq. (8.2) is equal to Q_D, the worst-case latency for the BT PDU is as follows:

$$t_{bt} = \sum_{i=1}^{ndp-1} q^i_\Delta + \sum_{j=1}^{ndp-1} \left(t^{j\to j+1}_{srBT} + t_{rd}\right) + C^{ndp}_{BT} - C^1_{BT} \quad (8.9)$$

8.4.3. Worst-case duration for the handoff procedure (t_{ho})

Consider again the Communication Network layout example depicted in Figure 8.1. Each SWLD (SWLD1, SWLD2, SWLD3) is associated to a different radio channel set (CH1, CH2, CH3). Figure 8.4 exemplifies a timing diagram for the mobility management procedure (queuing delays are not considered, as they are not relevant for the computation of the worst-case duration of the handoff procedure).

The worst-case duration of the handoff procedure (t_{ho}) is computed as follows. It is assumed that the mobile ES starts the handoff procedure immediately after receiving the BT PDU, beginning the assessment in the current radio channel (CH3, in the example). After that, the station switches to another radio channel (CH2) and does the assessment, switches to the other radio channel (CH1) and does the assessment, and finally switches to the radio channel with the best quality.

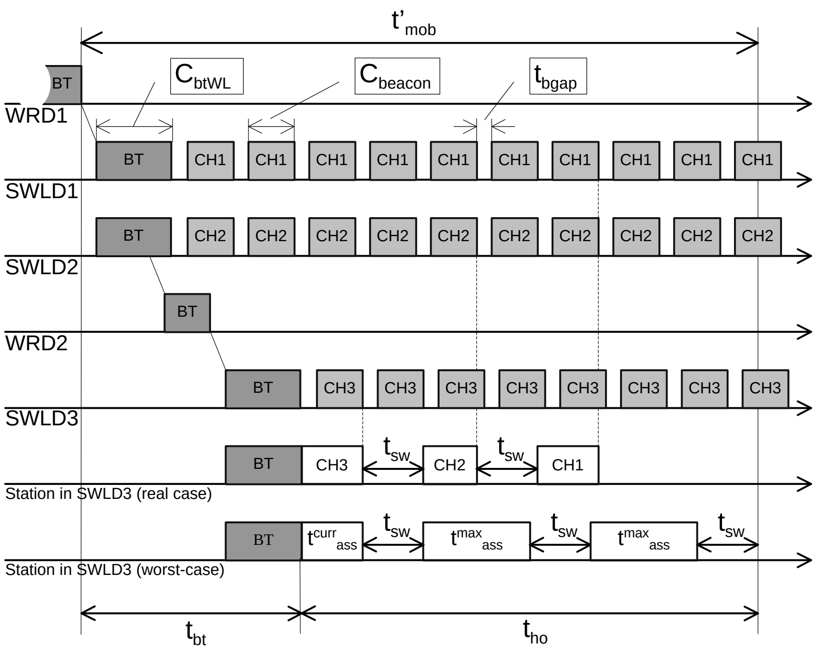


Figure 8.4: Mobility management timing diagram

The time elapsed in the assessment of the radio channel currently (t^{curr}_{ass} in Figure 8.4) being used by the mobile ES/LIS (CH3, in the example) is:

$$t_{ass}^{curr} = t_{bgap} + C_{beacon} \tag{8.10}$$

where C_{beacon} is the duration of a beacon (physical layer PDU) and t_{bgap} is the time interval between beacons.

As the MWLESs/MLISs in a certain SWLD start assessing the current radio channel immediately after the complete reception of the BT PDU, the correspondent SIS/SLIS starts transmitting beacons immediately after having (completely) relayed the BT PDU. As a result, the SIS/SLIS and the associated MWLESs/MLISs are synchronised, at the beginning of the beacon period. Therefore, in Eq. (8.10), t^{curr}_{ass} is just the sum of the beacon gap (t_{bgap}) and the duration of a beacon (C_{beacon}).

Oppositely, there is no synchronisation between a MWLES/MLIS and the other SISs/SLISs in the network, since these may receive the BT PDU at different instants. Considering the worst-case situation when assessing CH1 and CH2, i.e. the mobile station starts assessing the channel immediately after the beginning of the beacon. Therefore, the maximum assessment duration for each of those channels is:

$$t_{ass}^{\max} = 2 \cdot C_{beacon} + t_{bgap} \tag{8.11}$$

where C_{beacon} is the duration of a beacon (physical layer PDU) and t_{bgap} is the interval between beacons. Considering that the number of radio channels (to assess) is denoted as *nch* and the switching time is defined as t_{sw}, the maximum duration of the handoff procedure in the mobile ES is:

$$\begin{aligned} t_{ho} &= t_{ass}^{curr} + t_{sw} + (nch - 1) \cdot \left(t_{ass}^{\max} + t_{sw}\right) \Leftrightarrow \\ t_{ho} &= (2 \cdot nch - 1) \cdot C_{beacon} + nch \cdot \left(t_{bgap} + t_{sw}\right) \end{aligned} \tag{8.12}$$

In case three different radio channel sets are used (*nch=3*, as in the example presented), the worst-case duration of the handoff procedure will be:

$$t_{ho} = 5 \cdot C_{beacon} + 3 \cdot t_{bgap} + 3 \cdot t_{sw} \tag{8.13}$$

Therefore, the (preliminary) maximum duration of the mobility management period can be computed as (recalling Eq. (8.1)):

$$t'_{mob} = t_{bt} + t_{ho} \tag{8.14}$$

where t_{bt} and t_{ho} are given by Eqs. (8.2) and (8.12), respectively.

8.4.4. The number of beacons for each SIS/SLIS ($n_b(SLIS)$)

After having computed a preliminary value for the mobility management duration (t'_{mob}), it is now possible to determine the number of beacons that each SIS/SLIS must transmit. A SIS/SLIS starts transmitting beacons upon having received and (completely) relayed a BT PDU.

In order for the mobility management procedure to work properly, every SIS/SLIS must know the exact number of beacons they must transmit, which may vary depending on the SIS/SLIS. This is a parameter which is set in the SISs/SLISs. Moreover, considering that the beacon transmission is non pre-emptive (i.e., once a SIS/SLIS starts transmitting a beacon, it must complete the transmission until the end), there will be the need to adjust the mobility management period, as given by Eq. (8.14).

For every SWLD in the Communication Network, the following approach is then used:

1. compute the preliminary duration of the beacon period (t'_{bp});
2. compute the number of beacons that must be transmitted by the correspondent SIS/SLIS (n_b);
3. re-compute the beacon period duration (t_{bp}) for that SLWD;
4. compute the mobility management duration (t_{mob}) for that SWLD.

Obviously, the maximum value between the mobility management duration of all SISs/SLISs will be chosen as the mobility management duration for the network. This will be the value to be used to set the Idle Time parameter T_{ID2} for the MobM.

Computing the preliminary duration of the beacon period – t'$_{bp}$(SWLD)

The preliminary (maximum) duration of the beacon period, for a particular SWLD can be computed as follows:

$$t'_{bp}(SWLD) = t'_{mob} - t_{btn}(SWLD) \tag{8.15}$$

Figure 8.5 depicts the timing diagram for the case of SWLD1.

The reason why t_{btn} (latency of the BT PDU assuming no queuing) is used instead of t_{bt} (latency with queuing) is that a maximum value for the beacon period must be computed. This maximum value occurs for the lowest BT PDU latency, that is t_{btn} (as can be derived from Eq. (8.14)).

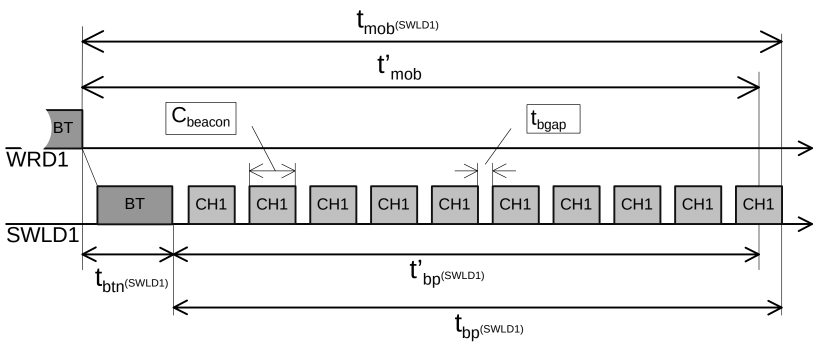


Figure 8.5: Example for the number of beacons (SWLD1)

Computing the number of beacons – n$_b$(SWLD)

The number of beacons that a particular SIS/SLIS must issue ($n_b(SWLD)$) will be:

$$n_b(SWLD) = \left\lceil \frac{t'_{bp}(SWLD)}{t_{bgap} + C_{beacon}} \right\rceil \tag{8.16}$$

For the case expressed in Figure 8.5, the number of beacons is 10.

Re-computing the beacon period duration (t$_{bp}$)

Now, it is necessary to re-compute the beacon period duration for the SWLD, taking into account the real number of beacons that must be issued by the corresponding SIS/SLIS, as given by Eq. (8.16):

$$t_{bp}(SWLD) = n_b(SWLD) \cdot \left(t_{bgap} + C_{beacon}\right) \tag{8.17}$$

From Figure 8.5, it is clear that $t_{bp}(SWLD1)$ is greater than $t'_{bp}(SWLD1)$.

Computing the mobility management duration – $t_{mob}(SWLD)$

As a consequence, the worst-case mobility management duration due to that SWLD must also be computed, i.e.:

$$t_{mob}(SWLD) = t_{bt}(SWLD) + t_{bp}(SWLD) \tag{8.18}$$

In this case, t_{bt} is considered, i.e. the latency of the BT PDU including the maximum queuing delay (Q), since a worst-case value for the mobility management duration is envisaged. Referring to the example depicted in Figure 8.5, it can be seen that $t_{mob}(SWLD1)$ is greater than the preliminary mobility management duration t'_{mob}.

8.4.5. Setting the Idle Time parameter (T_{ID2}) in the MobM

After having computed the mobility management duration for all SWLD (and the number of beacons for all the corresponding SISs/SLISs), it is necessary to determine the maximum between them, i.e.:

$$t_{mob} = \max\{t_{mob}(SWLD)\}, \quad \forall\, SWLD \text{ in the Communication Network} \tag{8.19}$$

Then, in the mobility master, the idle time parameter T_{ID2} should be set to a minimum value of:

$$T_{ID2} = \lceil t_{mob} \cdot r_{MobM} \rceil \tag{8.20}$$

where r_{MobM} represents the bit rate in the physical layer of the mobility master.

This idle time guarantees that there will be no collisions/jamming when the MobM resumes normal operation (transmits a token or a request PDU)

In Annex E, an algorithm for the computation of both T_{ID2} (for the MobM) and the number of beacons that each SIS/SLIS must transmit is presented.

8.4.6. Additional remarks

Location of the MobM

The mobility management duration depends on the location of the mobility master in the Communication Network. In order to optimise the performance/throughput of the Communication Network, this duration should be minimised at system design phase (pre-run-time). Therefore, the mobility management duration should be computed for different locations of the mobility master (Wired or Wireless Domains), in order to get its optimal (minimum) value. As a rule of thumb, the mobility master should not be located in “peripheral” Communication Domains, in order to “balance” the latency of the BT PDU from the MobM until the SISs/SLISs.

Priority of the BT PDU

Considering the PROFIBUS message dispatching algorithm (Figure 3.1), if a master ES receives a late token, it is only allowed to transmit one high priority message. Therefore, in order to guarantee the proper operation of the Communication Network, it is advisable to set the BT PDU as a high priority message (SDN) and to use a dedicated master ES as the MobM (no additional message streams in the master). This guarantees the transmission of the BT PDU upon reception of the token (and if the mobility management timer has expired).

8.5. Store&Forward behaviour

For the case where Intermediate Systems have a store&forward behaviour, all the analysis, formulae, results and algorithms presented in this Chapter are also valid. Again, the only difference is that the start relaying instant is expressed as $t^{i\text{®}j}_{sr} = C^i$ (Eq. 5.12).

A simplified approach where the IS act as store&forward repeaters and the Mobility Master is exclusively dedicated to mobility management was presented in (Alves et al., 2002). In that case, queuing delays never occur, due to the store&forward behaviour of the IS and to the fact that the length of the BT PDU is higher than the length of the token PDU. As a result, the computation of the mobility management parameters turns out to be simpler, since there is no need to compute Q (Q = 0).

8.6. Summary

Chapter 4 introduced an innovative mobility management mechanism that uses native PROFIBUS features and provides a seamless inter-cell mobility for mobile master and slave End Systems (MWLESs) and also for mobile Linking Intermediate Systems (MLISs, associated to MWRDs). Basically, a mobility manager (MobM) is responsible for starting a well-defined mobility management period, during which every Mobile End System or Mobile Linking Intermediate System will be able to perform the handoff procedure. One of the pros of this mechanism is the associated timing determinism, since the mobility management duration can be determined *a priori*.

This chapter mainly focused on the computation of the fundamental parameters of the Mobility Management procedure. One is the Idle Time parameter T_{ID2} in the Mobility Master (MobM), that guarantees that it waits a sufficient amount of time after transmitting the Beacon Trigger (BT) PDU and before transmitting another request or token PDU. The other parameter is the number of beacons that each Structuring Intermediate System (SIS) or Structuring & Linking Intermediate Systems (SLISs) must issue, after receiving the BT PDU. An algorithm for the computation of the mobility management parameters is provided in Annex E.

Chapter 9

Applying the Methodologies: Case Studies and Numerical Examples

In this chapter, some application scenarios are presented. The main objective is to demonstrate the applicability of the analysis carried out in Chapters 5, 6, 7 and 8. To this purpose, two case studies are elaborated. Additionally, some simulations are worked out with the objective of drawing some conclusions about the characteristics of the methodologies proposed in the previous chapters.

9.1. Introduction

This chapter has two main objectives: to exemplify how to apply the models described in Chapters 5, 6, 7 and 8 in practical cases, and to elaborate some conclusions on the proposed methodologies by analysing and comparing the numerical results. All the results for the case studies are obtained by using a system planning software application developed in the context of this thesis and which is based on the algorithms presented in Annexes B, D and E.

Figures are presented for the most relevant parameters, considering two different case studies. Case Study 1 addresses a not so complex Communication Network layout that does not support inter-cell mobility. The analytical models of the Communication Network, including the Communication Domains, Physical Media, End Systems, Intermediate Systems and Message Streams is also presented. Two different Physical Media are considered, for encompassing wired and wireless domains.

The Physical Medium of Wired Domains (WRDs) is assumed to be PROFIBUS RS-485 running at 1.5 Mbit/s. The Physical Medium of Wireless Domains (WLDs) is assumed to be based on the IEEE 802.11b physical medium (refer to §2.2.4 and §4.4.2), introducing some initial overhead in the physical layer PDU and with a bit rate of 2 Mbit/s. This overhead in the PhL PDU permits to fulfil some requirements (e.g., reliability) imposed by the wireless nature of the Physical Medium. The 2 Mbit/s bit rate is one of the bit rates encompassed by the IEEE 802.11b standard (IEEE 802.11b, 1999).

The Communication Network scenario presented in Case Study 2 differs essentially in the support of inter-cell mobility. In this case study, one of the master ESs assumes the Mobility Management functionality (the Mobility Master - *MobM*), three Intermediate Systems assume Structuring & Linking (SLIS) functionality (in order to have structured radio cells) and two ES are considered to be mobile.

9.2. Case Study 1: a network scenario without inter-cell mobility

In this case study, a Communication Network topology as depicted in Figure 9.1 will be considered.

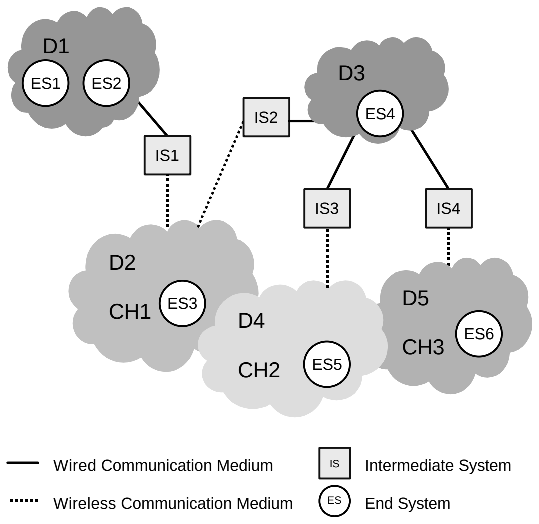


Figure 9.1: Network layout not considering inter-cell mobility

9.2.1. Communication Network model

In PROFIBUS, the number of data bits in each DLL char is eight (d = 8) and the length of the token PDU is three characters (L_{token} = 3). The maximum and minimum lengths for the request/response PDUs are considered, i.e. 255 characters maximum (L^{max}_{req} = L^{max}_{resp} = 255) and 6 characters minimum (L^{min}_{req} = L^{min}_{resp} = 6). In spite of the PROFIBUS short acknowledgement PDU being 1 character length, it will not be considered as the minimum length response, since it is rarely used (not used at all in PROFIBUS-DP) due to the non-existence of error detection (FCS field) in the PDU.

Concerning the parameters related to the physical layer, it is assumed that all Intermediate Systems have an internal relaying delay (t_{rd}) of 25 µs and that the minimum idle time (T_{IDm}) is equal to 100 bit times. The turnaround (reaction) time of all responders (either master or slave ESs) is assumed to be in the range of 10-50 µs (t^{min}_{rt} = 10 µs; t^{max}_{rt} = 50 µs).

Communication Domains and Physical Media

There are five Communication Domains (D) and two types of Physical Media (M) in the example Communication Network. Physical Medium M^1 is associated to Wired Domains

D^1 and D^3, while M^2 is associated to Wireless Domains D^2, D^4 and D^5. Therefore, the following sets can be defined:

Table 9.1: Sets of Communication Domains and Physical Media

$D = \{D^1, D^2, D^3, D^4, D^5\}$
$M = \{M^1, M^2\}$

with the Communication Domains parameters set as:

Table 9.2: Models of the Communication Domains

$D^1 = (WRD, M^1, \{IS^1\}, \{ES^1, ES^2\})$
$D^2 = (AWLD, M^2, \{IS^1, IS^2\}, \{ES^3\})$
$D^3 = (WRD, M^1, \{IS^2, IS^3, IS^4\}, \{ES^4\})$
$D^4 = (AWLD, M^2, \{IS^3\}, \{ES^5\})$
$D^5 = (AWLD, M^2, \{IS^4\}, \{ES^6\})$

The RS-485 physical layer version of PROFIBUS is assumed for wired communications. A (standard) bit rate of 1.5 Mbit/s ($r^{(1)} = 1.5$), no head or tail overhead bits ($l^1_H = l^1_T = 0$) and three additional bits per DLL char ($k^{(1)} = 3$), taking into account the UART character format (start, parity and stop bits) are considered (Figure 9.2).

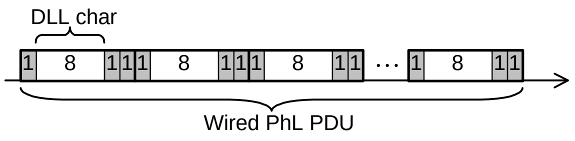


Figure 9.2: Format of a wired PhL PDU

When relaying a PhL PDU from a Wired Domain to a Wireless Domain, the Intermediate System removes every additional 3 bits and encapsulates the entire data octets in the data part of the wireless PhL PDU. The wireless PhL PDU also includes a preamble, start frame delimiter and header, that are usual in wireless communication physical layers, such as in IEEE 802.11b.

It is assumed that wireless communications run at 2 Mbit/s ($r^{(2)} = 2$), since it is a standard bit rate in IEEE 802.11b. The wireless physical layer PDU (Figure 9.3) is assumed to have an overhead of 200 bits ($l^2_H = 200$), corresponding to preamble, start frame delimiter and header information. The tail overhead is considered to be null ($l^2_T = 0$). Moreover, it is considered that the wireless physical layer does not introduce any overhead per DLL char ($k^{(2)} = 0$). Therefore, the Physical Media can be defined as :

Table 9.3: Model of the Physical Media

$M^1 = (1.5, 0, 0, 3)$ and $o^1 = 33$
$M^2 = (2, 200, 0, 0)$ and $o^2 = 150$

where the offset (that represents the total number of bits since the beginning of the PhL PDU until the length of the data field is known (*o*)), is different for the two physical mediums, as it is explained next.

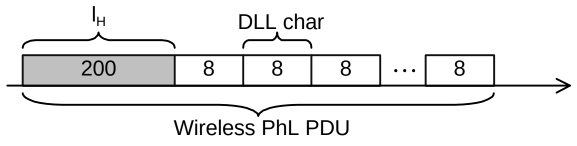


Figure 9.3: Format of a wireless PhL PDU

In the case of the wired physical medium (PROFIBUS RS-485), it must be taken into account that the "length of data" information is found inside the DLL PDU either in an implicit or explicit way, depending on the type of DLL PDU. In all types of PDUs but the "variable length" PDU type, the length is implicit to the Start Delimiter (SD) field, since there is a unique identifier (SD1-4) for each PDU type (e.g. SD3 corresponds to a "Fixed length frame with data field" – refer to Chapter 3). Since the Start Delimiter field is always the first field of the DLL PDU, the offset (o^l) for this type of PDU is always equal to 11 bits (1 UART char). Nevertheless, for the case of the variable data field type of PDU (with Start Delimiter SD2), the length of the DLL PDU is explicitly present at the beginning of the DLL PDU (*LE*,*LEr*). Therefore, the offset (o^l) would depend on the type of PROFIBUS DLL PDU being considered. For the sake of simplicity, it has been considered that the offset for wired physical media (Figure 9.4) is always equal to 33 bits ($o^{(1)} = 33$), i.e. the length of the DLL PDU is only known at the end of the third character (after the *LEr* field).

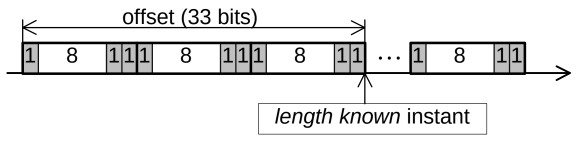


Figure 9.4: Offset of wired PhL PDUs

For the wireless physical medium (Figure 9.5), it is assumed that the length of the data field is included in the PhL PDU header (as in the IEEE 802.11b protocol).

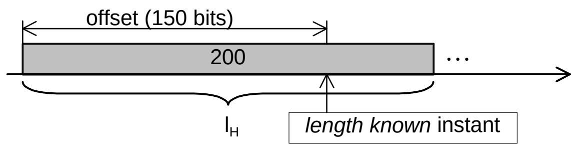


Figure 9.5: Offset of wireless PhL PDUs

Therefore, the length of the DLL PDU is known before the PhL data field itself. The wireless PhL offset is assumed to be 150 bits ($o^{(2)} = 150$, as presented in Table 9.3).

End Systems and Intermediate Systems

There is a total of six End Systems (*ES*) and four Intermediate Systems (*IS*) in the Communication Network example (Table 9.4). Three of the End-Systems are wired (ES^1, ES^2 and ES^4) and other three are wireless (ES^1, ES^2 and ES^4). Since it is assumed that the Communication Network does not support inter-cell mobility, all the Intermediate Systems are considered to be Linking Intermediate Systems (LISs).

Table 9.4: Sets of End Systems and Intermediate Systems

$$ES = \{ES^1, ES^2, ES^3, ES^4, ES^5, ES^6, ES^7\}$$
$$IS = \{IS^1, IS^2, IS^3, IS^4\}$$

Each of the End Systems is characterised as in Table 9.5.

Table 9.5: Models for the End Systems

$ES^1 = (WRES, M^1, MASTER)$	$ES^2 = (WRES, M^1, SLAVE)$
$ES^3 = (WLES, M^2, SLAVE)$	$ES^4 = (WRES, M^1, SLAVE)$
$ES^5 = (WLES, M^2, MASTER)$	$ES^6 = (WLES, M^2, SLAVE)$

Considering that the relaying delay (t_{rd}) and the minimum idle time (T_{IDm}) parameters are common for all Intermediate Systems, these are characterised as illustrated in Table 9.6.

Table 9.6: Models for the Intermediate Systems

$IS^1 = (LIS, \{M^1, M^2\}, -, -)$	$IS^2 = (LIS, \{M^1, M^2\}, -, -)$
$IS^3 = (LIS, \{M^1, M^2\}, -, -)$	$IS^4 = (LIS, \{M^1, M^2\}, -, -)$

Message Streams

Consider eighteen message streams in the network ($S = \{S^1, \ldots, S^{18}\}$), which are defined by the parameters as presented in Table 9.7. Note that the message streams' *Initiator* and *Responder* attributes are defined in a way that it is possible to show the impact of the number of Intermediate Systems between initiator and responder on the system turnaround time and duration of message transactions (and Slot Time parameter).

Table 9.7: Models for the Message Streams

Message Stream	Initiator ES	Responder ES	L_{req} (chars)	L_{resp} (chars)
S^1	ES^1	ES^2	255	6
S^2	ES^1	ES^2	59	59
S^3	ES^1	ES^2	6	255
S^4	ES^1	ES^3	255	6
S^5	ES^1	ES^3	59	59
S^6	ES^1	ES^3	6	255
S^7	ES^1	ES^4	255	6
S^8	ES^1	ES^4	59	59
S^9	ES^1	ES^4	6	255
S^{10}	ES^1	ES^6	255	6
S^{11}	ES^1	ES^6	59	59
S^{12}	ES^1	ES^6	6	255

S^{13}	ES^5	ES^4	255	6
S^{14}	ES^5	ES^4	59	59
S^{15}	ES^5	ES^4	6	255
S^{16}	ES^5	ES^2	255	6
S^{17}	ES^5	ES^2	59	59
S^{18}	ES^5	ES^2	6	255

For each initiator/responder pair, 3 different request/response lengths are defined: long request (255 chars) with short response (6 chars), medium length request (59 chars) with medium length response (59 chars) and short request (6 chars) with long response (255 chars). This diversity in PDU length will also influence the computation of the system turnaround time and duration of message transactions (and Slot Time parameter). Moreover, it will be shown that the worst-case queuing delay (Q) affecting a request PDU is strongly influenced by the PDU's length.

9.2.2. Duration of wired and wireless PhL PDUs

The duration of wired and wireless PhL PDUs can be computed using Eq. (5.4), as follows:

$$C^{wr} = \frac{L \times 11}{1,5} = \frac{22}{3} \times L \quad (ms) \quad \text{and} \quad C^{wl} = \frac{200 + L \times 8}{2} = 4 \times L + 100 \quad (ms)$$

Table 9.8 presents the PhL PDU duration for several PROFIBUS PDU lengths.

Table 9.8: DLL PDU length and PhL PDU duration

PDU Type	L (chars)	C^{wr} (μs)	C^{wl} (μs)
Short acknowledge	1	7.3	104
Token	3	22	112
Fixed length no data	6	44	124
1 data octet (BT)	10	73.3	140
50 data octets	59	432.7	336
100 data octets	109	799.3	536
150 data octets	159	1166	736
246 data octets	255	1870	1120

It is clear that shorter PDUs have a longer duration in Wireless Domains (due to the additional overhead in the PhL PDU) while longer PDUs take a longer duration to be transmitted in Wired Domains (due to lower bit rate). As it will be shown, this fact has a strong impact in some relevant parameters, namely the Idle Times.

9.2.3. Start relaying instant

To evaluate the start relaying instant (Eq. (5.8)), it is necessary to compute the *data ready* (t^i_{dr}), *length known* (t^i_{lk}) and *no gap* ($t^{i®j}_{ng}$) instants, assuming that Communication Domain i is wired and Communication Domain j is wireless and vice-versa.

Start relaying instant from a WRD to a WLD

The *data ready* instant for a PDU arriving from a Wired Domain ($t^{wr}{}_{dr}$) can be computed using Eq. (5.9), as follows:

$$t_{dr}^{wr} = \frac{0+3+8}{1.5} = 7.33\, m\mathrm{s}$$

The *length known* instant for a PDU arriving from a Wired Domain ($t^{wr}{}_{lk}$) can be computed using Eq. (5.10), as follows:

$$t_{lk}^{wr} = \frac{33}{1.5} = 22\, m\mathrm{s}$$

The *no gap* instant for a PDU arriving from a Wired Domain to a Wireless Domain ($t^{wr®wl}{}_{ng}$) can be computed using Eq. (5.11), as follows:

$$t_{ng}^{wr \to wl} = \frac{0}{1.5} - \frac{200}{2} + L \times \left(\frac{8+3}{1.5} - \frac{8+0}{2} \right) - \frac{8+0}{2} = \frac{10}{3} \times L - 104$$

As a consequence, for $L \leq 37$, the *start relaying* instant is equal to the *length known* instant, i.e. 22 μs. Nevertheless, for $L \geq 38$, the *start relaying* instant is equal to the *no gap* instant (please refer to Eq. (5.8)). Therefore, the *start relaying* instant for the case of PDUs being relayed from a WRD to a WLD will be as illustrated in Table 9.9.

Table 9.9: Start relaying instant from WRD to WLD

PDU Type	L (chars)	t_{sr} (μs)
Token	3	22
Fixed length no data	6	22
50 data octets	59	92.7
246 data octets	255	746.0

Start relaying instant from a WLD to a WRD

The *data ready* instant for a PDU arriving from a Wireless Domain ($t^{wl}{}_{dr}$) can be computed using Eq. (5.9), as follows:

$$t_{dr}^{wl} = \frac{200+8}{2} = 104\, m\mathrm{s}$$

The *length known* instant for a PDU arriving from a Wireless Domain ($t^{wl}{}_{lk}$) can be computed using Eq. (5.10), as follows:

$$t_{lk}^{wl} = \frac{150}{2} = 75\, m\mathrm{s}$$

The *no gap* instant for a PDU arriving from a Wireless Domain to a Wired Domain ($t^{wl®wr}{}_{ng}$) can be computed using Eq. (5.11), as follows:

$$t_{ng}^{wl \to wr} = \frac{200}{2} - \frac{0}{1.5} + L \times \left(\frac{8+0}{2} - \frac{8+3}{1.5} \right) - \frac{8+3}{1.5} = \frac{278}{3} - \frac{10}{3} \times L$$

As a consequence, the *start relaying* instant (Eq. (5.8)) for the case of a PDU being relayed from a WLD to a WRD is always equal to the *data ready* instant (104 μs), independently of the DLL PDU length (*L*).

9.2.4. Computation of T_{ID1} and T_{ID2}

Computation approach

The idle time parameters were computed using the (simplified) algorithm presented in Annex B, which does not take into account the particular set of message streams for each master ES. Therefore, there will be only one value of T_{ID1} and only one value of T_{ID2} for all master ESs belonging to Wired Domain (ES^1, in this case study), since all Wired Domains are associated to the same Physical Medium (M^1). The same happens with wireless ESs, i.e. all master ESs belonging to Wireless Domains (ES^5, in this case study) set the same value for T_{ID1} and the same value for T_{ID2}, since all Wireless Domains are associated to the same Physical Medium (M^2).

Computation of t_{ID1+} and T_{ID1} for WRDs

Since there are only two types of Physical Media (M^1 and M^2), the idle time for ES belonging to WRDs must be evaluated just for the case where the other Communication Domain is wireless. The adequate request and response DLL PDU lengths must be chosen to maximise the inserted idle time. Applying Eq. (6.17) to this case stands:

$$\begin{cases} \dfrac{d+k^j}{r^j} = \dfrac{d+k^{wl}}{r^{wl}} = \dfrac{8+0}{2} = 4 \\ \dfrac{d+k^i}{r^i} = \dfrac{d+k^{wr}}{r^{wr}} = \dfrac{8+3}{1.5} = 7.(3) \end{cases} \Rightarrow \dfrac{d+k^j}{r^j} < \dfrac{d+k^i}{r^i}$$

This means that the duration of a character is smaller in the wireless medium (*j*) than in the wired medium (*i*). Therefore, for the computation of the idle time for ES belonging to WRD, the following DLL PDU lengths must be considered:

$$L_{req(l-1)} = L_{req}^{\min}, L_{resp(l-1)} = L_{resp}^{\min}, L_{req(l)} = L_{token}$$

Then, $t^{wr®wl}_{ID1+}$ must be evaluated as the maximum between $t^{wr®wl}_{ID1G+}$ (Eq. (6.6)) and $t^{wr®wl}_{IDD1+}$ (Eq. (6.21)), where:

$$t_{ID1\Gamma+}^{wr\rightarrow wl} \geq 124-44+124-44+2\times\frac{100}{2}-\frac{100}{1.5}-10+22-22+$$
$$+\max\left\{22-22+44-124+10-\frac{100}{2},\ 0\right\} = 183.(3)\ ms$$

$$t_{ID1\Delta+}^{wr\rightarrow wl} \geq 22-22+112-22+\frac{100}{2}-\frac{100}{1.5} = 73.(3)\ ms$$

Therefore, the inserted idle time after receiving a response PDU or receiving the token PDU is $t^{wr®wl}_{ID1+}$ = 183.(3) µs. The T_{ID1} parameter can be computed using Eq. (6.28), as follows:

$$T_{ID1}^{wr} = 100 + \lceil 183.(3) \times 1.5 \rceil = 375\,\text{bits}$$

Computation of t_{ID2+} and T_{ID2} for WRDs

The same DLL PDU lengths as for the previous sub-section must be considered, in order to get the worst-case inserted idle time required after issuing an unacknowledged request PDU. Therefore, $t^{wr®wl}_{ID2+}$ can be computed using Eq. (6.32) as follows:

$$t_{ID2+}^{wr \to wl} \geq 22 - 22 + 124 - 44 + \frac{100}{2} - \frac{100}{1.5} = 63.(3)\ m\text{s}$$

which leads to a T_{ID2} parameter (Eq. (6.39)) equal to:

$$T_{ID2}^{wr} = 100 + \lceil 63.(3) \times 1.5 \rceil = 195\,\text{bits}$$

Computation of t_{ID1+} and T_{ID1} for WLDs

As there are only two Physical Media in the Communication Network, the idle time for ESs belonging to WLDs must be evaluated just for the case where the other Communication Domain is wired.

The adequate request and response DLL PDU lengths must be chosen as follows, to determine the required inserted idle time (Eq. (6.17)):

$$\begin{cases} \dfrac{d + k^j}{r^j} = \dfrac{d + k^{wr}}{r^{wr}} = \dfrac{8+3}{1.5} = 7.(3) \\ \dfrac{d + k^i}{r^i} = \dfrac{d + k^{wl}}{r^{wl}} = \dfrac{8+0}{2} = 4 \end{cases} \Rightarrow \frac{d + k^j}{r^j} > \frac{d + k^i}{r^i}$$

Therefore, the following DLL PDU lengths must be considered:

$$L_{req(l-1)} = L_{req}^{\max}, L_{resp(l-1)} = L_{resp}^{\max}, L_{req(l)} = L_{req}^{\max}$$

Note that for this case, the lengths of the DLL PDUs to be considered are the opposite (maximum) to those considered for ESs belonging to WRDs.

Then, $t^{wl®wr}_{ID1+}$ is the maximum between $t^{wl®wr}_{ID1G+}$ (Eq. (6.6)) and $t^{wl®wr}_{IDD1+}$ (Eq. (6.21)), where:

$$\begin{cases} t_{ID1\Gamma+}^{wl \to wr} \geq 1870 - 1120 + 1870 - 1120 + 2 \times \dfrac{100}{1.5} - \dfrac{100}{2} - 10 + 104 - 104 + \\ \qquad + \max\left\{ 104 - 104 + 1120 - 1870 + 10 - \dfrac{100}{1.5},\ 0 \right\} = 1573.(3)\ m\text{s} \\ t_{ID1\Delta+}^{wl \to wr} \geq 104 - 104 + 44 - 124 + \dfrac{100}{1.5} - \dfrac{100}{2} = -63.(3)\ m\text{s} \end{cases}$$

Therefore, the inserted idle time after receiving a response PDU or receiving the token PDU is $t^{wl®wr}_{ID1+}$ = 1573.(3) μs, and T_{ID1} (Eq. (6.28)) must be set to:

$$T^{wl}_{ID1} = 100 + \lceil 1573.(3) \times 2 \rceil = 3247 \text{ bits}$$

Computation of t_{ID2+} and T_{ID2} for WLDs

The same DLL PDU lengths, as for the previous sub-section, must be considered, in order to obtain the worst-case the inserted idle time required after issuing an unacknowledged request PDU. Therefore, $t^{wl®wr}_{ID2+}$ can be computed using Eq. (6.32) as follows:

$$t^{wl\rightarrow wr}_{ID2+} \geq 104 - 104 + 1870 - 1120 + \frac{100}{1.5} - \frac{100}{2} = 766.(6)\, \mathit{m}\text{s}$$

which leads to a T_{ID2} parameter (Eq. (6.39)) equal to:

$$T^{wl}_{ID2} = 100 + \lceil 766.(6) \times 2 \rceil = 1634 \text{ bits}$$

Summarising Table for the Idle Time values

Table 9.10 summarises the idle time values for this case study, which were computed using the software tool implementing the algorithm described in Annex B.

Table 9.10: Idle time values

ES Type	t_{ID1+} (μs)	T_{ID1} (bits)	t_{ID2+} (μs)	T_{ID2} (bits)
Wired Master	183.33	375	63.33	195
Wireless Master	1573.33	3247	766.67	1634
Intermediate System	-	100	-	100

As expected, the idle times inserted by wired ESs (183.3 μs and 63.3 μs) are much smaller than the idle times inserted by wireless ESs (1573.3 μs and 766.7 μs). The reason for this is that the inserted idle time for the wired ESs is computed considering the minimum DLL PDU lengths, while the inserted idle time for the wireless ESs is computed considering the maximum DLL PDU lengths.

9.2.5. Computation of t_{st} and T_{SL1}

Table 9.11 summarises the worst-case system turnaround time (t_{st}) and duration of message transactions (C_{ack}) obtained by applying the methodologies described in the previous sections for Case Study 1. All parameters were computed using the software tools implementing the algorithms described in Annexes B and D.

The *Path* column describes the Physical Media in the path from initiator to responder. For instance, the *path* for message stream 4 (S^4) is $\{M^1, M^2\}$, since the request PDU is issued from D^1 (which is associated to M^1), is relayed by IS^1 and arrives to the responder at D^2 (which is associated to M^2). For the sake of space, the Physical Media will be denoted by the number (‘M^1‘ is represented as ‘1’ and ‘M^2‘ is represented as ‘2’).

Table 9.11: System turnaround times and duration of message transactions

Message Stream	Initiator ES	Responder ES	L_{req}/L_{resp} (chars)	Path (M,…)	t_{stn} (μs)	Q (μs)	t_{st} (μs)	C_{ack} (μs)
S^1	ES^1	ES^2	255/6	1	50	0	**50$^{(a)}$**	**2214$^{(b)}$**
S^2	ES^1	ES^2	59/59	1	50	0	50$^{(a)}$	1165
S^3	ES^1	ES^2	6/255	1	50	0	50$^{(a)}$	2214
S^4	ES^1	ES^3	255/6	1,2	200	0	200	2364
S^5	ES^1	ES^3	59/59	1,2	200	0	200	1315.3
S^6	ES^1	ES^3	6/255	1,2	306	0	306	2470
S^7	ES^1	ES^4	255/6	1,2,1	1126	0	1126	3290
S^8	ES^1	ES^4	59/59	1,2,1	543.3	**590$^{(d)}$**	1133.3	2248.7
S^9	ES^1	ES^4	6/255	1,2,1	1126	660.7	1786.7	3950.7
S^{10}	ES^1	ES^6	255/6	1,2,1,2	1276	0	1276	3440
S^{11}	ES^1	ES^6	59/59	1,2,1,2	693.3	590	1283.3	2398.7
S^{12}	ES^1	ES^6	6/255	1,2,1,2	1382	660.7	2042.7	4206.7
S^{13}	ES^5	ES^4	255/6	2,1	976	0	976	3843.5
S^{14}	ES^5	ES^4	59/59	2,1	393.3	0	393.3	2688.3
S^{15}	ES^5	ES^4	6/255	2,1	**870$^{(c)}$**	0	870	3737.5
S^{16}	ES^5	ES^2	255/6	2,1,2,1	2052	0	2052	4919.5
S^{17}	ES^5	ES^2	59/59	2,1,2,1	886.7	653.2	1539.9	3835.3
S^{18}	ES^5	ES^2	6/255	2,1,2,1	1946	723.9	**2669.9**	5537.3

The first component of the Slot Time parameter – t_{SL1} – should be greater than the maximum between the worst-case system turnaround time of all message transactions in the Communication Network, i.e. t_{SL1} = 2669.9 μs (dashed cell in Table 9.11).

There are some remarks concerning the results presented in Table 9.11 which are worthwhile to go through.

Message Streams with no IS between initiator and responder

In the case where the initiator and the responder communicate without any IS in the path between them, the worst-case system turnaround time is equal to the maximum responders’ turnaround time (referenced with an (a), in Table 9.11):

$$t_{st} = t_{rt}^{\max} = 50\,m\mathrm{s}$$

This is the case of Message Streams 1, 2 and 3. Concerning the duration of these message transactions, they may be computed using Eq. (7.23). As an example, message transactions corresponding to Message Stream 1 have the following worst-case duration (referenced with a (b), in Table 9.11):

$$C_{ack} = 1870 + 50 + 44 + \frac{100}{1.5} + 183.3 = 2214\,m\mathrm{s}$$

Message Streams with one IS between initiator and responder

By definition, the inserted idle time guarantees that a request PDU is affected by no queuing delays in the first IS between initiator and responder. Therefore, for message transactions corresponding to Message Streams 4, 5, 6, 13, 14 and 15, it is guaranteed that *Q = 0*. However, the worst-case system turnaround time is no longer equal to the maximum responders' turnaround time, since both request and response PDUs are relayed by one IS, i.e. there is an additional latency that must be considered.

Consider Message Stream 15, for example. The worst-case system turnaround time can be computed using Eq. (7.1) and assuming *ndp* = 2 (*ndp* is the number of Communication Domains between initiator and responder):

$$\begin{aligned} t_{stn}^{1\to ndp} &= t_{srreq}^{1\to 2} + t_{rd} + C_{req}^{2} + t_{rt}^{\max} + t_{srresp}^{2\to 1} + t_{rd} - C_{req}^{1} \\ &= 104 + 25 + 44 + 50 + 746 + 25 - 124 = 870\, ms \end{aligned}$$

This result is referenced with a (c), in Table 9.11.

Message Streams with two or more ISs between initiator and responder

When there are two or more ISs between initiator and responder of a message stream, the request PDU of the correspondent message transaction may be affected by queuing delays. Therefore, the worst-case total queuing delay *Q* must be computed, in order to determine the worst-case system turnaround time for that message transaction. As an example, it will be demonstrated how the worst-case queuing delay *Q* is computed, for message transactions correspondent to *S8*.

First, it is necessary to evaluate *Qgama* (denoted as Q_G, in §7.3.4), the worst-case total queuing delay assuming that the previous transaction is an acknowledged (request/response) transaction. For that purpose, it is necessary to compute the worst-case queuing delays (*q_Gama* – denoted as q_G, in §7.3.4) in every IS between initiator (ES^1) and responder (ES^4). For the first IS, *Gama_a*[1] and *Gama_b*[1] (denoted as G^1_a and G^1_b, in §7.3.4) can be computed (Eq. (7.12)), as follows:

$$\begin{cases} Gama\_a[1] = 1870 + 10 + 1870 + \dfrac{100}{1.5} + 183.3 + 92.7 + 25 = 4117.7\, m \\ Gama\_b[1] = \max\left\{1870 + 10 + 746 + 25,\ 746 + 25 + 1120 + \dfrac{100}{2}\right\} + 1120 + \dfrac{100}{2} \\ \qquad\qquad = \max\{2651,\ 1941\} + 1170 = 3821\, ms \end{cases}$$

Since *Gama_a*[1] is greater than *Gama_b*[1], there is no queuing delay in the first IS (*q_Gama*[1] = 0), as expected.

For the second IS, the worst-case queuing delays *Gama_a*[2] and *Gama_b*[2] can be computed (Eq. (7.13 or 7.14)) as follows:

$$\begin{cases} Gama\_a[2] = \max\{4117.7,\ 3821\} + 104 + 25 = 4246.7\ m\text{s} \\ Gama\_b[2] = \max\begin{Bmatrix} 3821 - 1120 - \dfrac{100}{2} + 104 + 25, \\ 746 + 25 + 104 + 25 + 1870 + \dfrac{100}{1.5} \end{Bmatrix} + 1870 + \dfrac{100}{1.5} \\ \qquad = \max\{2780,\ 2836.7\} + 1936.7 = 4773.3\ m\text{s} \end{cases}$$

In this case, *Gama_b*[2] is greater than *Gama_a*[2], resulting in a worst-case queuing delay of *q_Gama*[2] = *Gama_b*[2] - *Gama_a*[2] = 4773.3 – 4246.7 = 526.6 μs.

For the third IS, *Gama_a*[3] and *Gama_b*[3] result in (Eq. (7.14)):

$$\begin{cases} Gama\_a[3] = \max\{4225.6,\ 4773.3\} + 92.7 + 25 = 4891\ m\text{s} \\ Gama\_b[3] = \max\begin{Bmatrix} 4773.3 - 1870 - \dfrac{100}{1.5} + 746 + 25, \\ 746 + 25 + 104 + 25 + 746 + 25 + 1120 + \dfrac{100}{2} \end{Bmatrix} + 1120 + \dfrac{100}{2} \\ \qquad = \max\{3607.6,\ 2841\} + 1170 = 4777.6\ m\text{s} \end{cases}$$

Since *Gama_b*[3] is smaller than *Gama_a*[3], there is no queuing delay in the third IS. Then, the sum the worst-case queuing delays in all ISs gives the worst-case total queuing delay that is equal to 526.6 μs (*Qgama* = 526.6 μs).

Now, it is necessary to compute the worst-case total queuing delay considering that the previous transaction is an unacknowledged request (*Qfi* – denoted as Q_F, in §7.3.5). Thus, it is necessary to compute the worst-case queuing delays (*q_Fi* – denoted as q_F, in §7.3.5) in every IS between initiator (ES^1) and responder (ES^4). For the first IS, *Fi_a*[1] and *Fi_b*[1] (denoted as F^1_a and F^1_b, in §7.3.5) can be computed using Eq. (7.17):

$$\begin{cases} Fi\_a[1] = 1870 + \dfrac{100}{1.5} + 63.3 + 92.7 + 25 = 2117.7\ m\text{s} \\ Fi\_b[1] = 746 + 25 + 1120 + \dfrac{100}{2} = 1941\ m\text{s} \end{cases}$$

Since *Fi_a*[1] is greater than *Fi_b*[1], there is no queuing delay in the first IS (*q_Fi*[1] = 0), as it was expected. For the second IS, *Fi_a*[2] and *Fi_b*[2] can be computed (Eq. (7.18 or 7.19)), as follows :

$$\begin{cases} Fi\_a[2] = \max\{2117.7, 1941\} + 104 + 25 = 2246.7\ m\text{s} \\ Fi\_b[2] = 746 + 25 + 104 + 25 + 1870 + \dfrac{100}{1.5} = 2836.7\ m\text{s} \end{cases}$$

In this case, *Fi_b*[2] is greater than *Fi_a*[2], resulting in a queuing delay of *q_Fi*[2] = *Fi_b*[2] - *Fi_a*[2] = 2836.7 – 2246.7 = 590 μs.

Finally, for the third IS, *Fi_a*[3] and *Fi_b*[3] can be computed using Eq. (7.19), as follows:

$$\begin{cases} Fi\_a[3] = \max\{2246.7,\ 2836.7\} + 96.7 + 25 = 2958.3\, m\text{s} \\ Fi\_b[3] = 746 + 25 + 104 + 25 + 746 + 25 + 1120 + \dfrac{100}{2} = 2841\, m\text{s} \end{cases}$$

Since *Fi_b*[3] is smaller than *Fi_a*[3], there is no queuing delay in the third IS. Then, the worst-case total queuing delay is equal to the sum the worst-case queuing delays in all ISs, resulting in *Qfi* = 590 μs.

Taking the values of *Qdelta* and *Qfi* into account, the worst-case total queuing *Q* can be determined using Eq. (7.22) (referenced with a (d), in Table 9.11):

$$Q = \max\{526.6,\ 590\} = 590\, m\text{s}$$

9.2.6. Computation of T_{SL2} and T_{SL}

Table 9.12 summarises the most relevant parameters for the computation of the worst-case system turnaround time after a master ES passing the token to its successor. The *Path* column describes the Physical Media in the path of the token PDU. The second component of the Slot Time parameter – t_{SL2} – should be greater than the maximum system turnaround time after passing the token, considering every possible token passing in the Communication Network, i.e. t_{SL2} = 3626.2 μs (dashed cell in Table 9.12).

Table 9.12: Parameters for the computation of T_{SL2}

Token passing	Path (M,…)	Sum_ $t_{srtoken}$ (μs)	Q (μs)	t^{ndp}_{ID1} (μs)	MaxSum _ t_{sr} (μs)	C^{ndp}_{token} (μs)	C^{1}_{token} (μs)	t_{st_token} (μs)
ES^1 ® ES^5	1,2,1,2	223	660.7	1623.5	1029	112	22	**3626.2**
ES^5 ® ES^1	2,1,2,1	305(a)	724(b)	250(c)	1671(d)	22(e)	112(f)	2860(g)

The slot time t_{SL} can be computed as the maximum between t_{SL1} and t_{SL2} (Eq. (7.25)):

$$t_{SL} = \max\{2669.8,\ 3626.2\} = 3626.2\, m\text{s}$$

Eq. (7.26) permits to compute the value of the slot time in bit times (T_{SL}), for every master ES in the Communication Network. In the case study, ES^1 belongs to a WRD and ES^5 to a WLD. In the case of master ES belonging to WRD (e.g ES^1), the Slot Time parameter must be set to:

$$T_{SL}^{wr} = \lceil t_{SL} \cdot r^{wr} \rceil = \lceil 3626.2 \times 1.5 \rceil = 5440 \text{ bits}$$

In the case of master ESs belonging to WLDs (e.g., ES^5), the Slot Time parameter must be set to:

$$T_{SL}^{wl} = \lceil t_{SL} \cdot r^{wl} \rceil = \lceil 3626.2 \times 2 \rceil = 7253 \text{ bits}$$

Note that the codification error referred in §7.5.1 (5440/1.5 ≠ 7253/2) will turn the slot time slightly different for WRES and WLES, but there is no impact on network operation.

All the relevant parameters for the case where the token is passed from ES^5 to ES^1 are computed next.

Computation of the latency of the token PDU

The value referenced with (a) in Table 9.12 represents the time for relaying the token PDU along the path between ES^5 and ES^1, and can be computed using Eq. (7.29) as follows (ndp = 4):

$$Sum\_t_{srLtoken} = 104 + 25 + 22 + 25 + 104 + 25 = 305\,\mu s$$

Computation of the worst-case total queuing delay affecting the token PDU

Since there are three IS between ES^5 and ES^1, there may be a queuing delay affecting the token PDU, thus Q must be computed. First, it is necessary to compute $Qgama$, the worst-case total queuing delay assuming the previous transaction is an acknowledged (request/response) transaction and assuming that the request PDU of transaction l is the token PDU. For that purpose, the worst-case queuing delays (q_Gama) in every IS between ES^5 and ES^1 must be computed.

For the first IS, $Gama\_a$[1] and $Gama\_b$[1] result in (Eq. (7.12)):

$$\begin{cases} Gama\_a[1] = 1120 + 10 + 1120 + \dfrac{100}{2} + 1573.3 + 104 + 25 = 4002.3\,\mu s \\ Gama\_b[1] = \max\left\{1120 + 10 + 104 + 25,\ 104 + 25 + 1870 + \dfrac{100}{1.5}\right\} + 1870 + \dfrac{100}{1.5} \\ \qquad = \max\{1259,\ 2065.(6)\} + 1936.(6) = 4002.(3)\,\mu s \end{cases}$$

Since $Gama\_a$[1] is equal to $Gama\_b$[1], there is no queuing delay in the first IS (q_Gama[1] = 0), as it was expected.

For the second IS, $Gama\_a$[2] and $Gama\_b$[2] can be computed using Eq. (7.13 or 7.14), as:

$$\begin{cases} Gama\_a[2] = \max\{4002.3,\ 4002.3\} + 22 + 25 = 4049.3\,\mu s \\ Gama\_b[2] = \max\left\{\begin{matrix} 4002.3 - 1870 - \dfrac{100}{1.5} + 746 + 25, \\ 104 + 25 + 746 + 25 + 1120 + \dfrac{100}{2} \end{matrix}\right\} + 1120 + \dfrac{100}{2} \\ \qquad = \max\{2836.6,\ 2070\} + 1170 = 4006.3\,\mu s \end{cases}$$

No queuing delay occurs in the second IS, since $Gama\_a$[2] is greater than $Gama\_b$[2].

Finally, for the third IS, *Gama_a*[3] and *Gama_b*[3] result in (Eq. (7.14)):

$$\begin{cases} Gama\_a[3] = \max\{4049.3,\ 4006.3\} + 104 + 25 = 4178.(3)\ m\text{s} \\ Gama\_b[3] = \max\left\{\begin{matrix} 4006.3 - 1120 - \dfrac{100}{2} + 104 + 25, \\ 104 + 25 + 746 + 25 + 104 + 25 + 1870 + \dfrac{100}{1.5} \end{matrix}\right\} + 1870 + \dfrac{100}{1.5} \\ \qquad = \max\{2965.3,\ 2965.7\} + 1936.7 = 4902.(3)\ m\text{s} \end{cases}$$

In this case, *Gama_b*[3] is greater than *Gama_a*[3], resulting in a worst-case queuing delay of *q_Gama*[3] = *Gama_b*[3] - *Gama_a*[3] = 4902.(3) – 4178.(3) = 724 μs. Finally, the sum the worst-case queuing delays in all ISs gives the worst-case total queuing delay, which in this case will be 724 μs (*Qgama* = 724 μs).

Now, it is necessary to compute the worst-case total queuing delay considering that the previous transaction is an unacknowledged request (*Qfi*) and the request PDU of transaction *l* is the token PDU. Thus, it is necessary to compute the worst-case queuing delays (*q_Fi*) in every IS between ES^5 and responder ES^1. For the first IS, *Fi_a*[1] and *Fi_b*[1] can be computed using Eq. (7.17), as:

$$\begin{cases} Fi\_a[1] = 1120 + \dfrac{100}{2} + 766.7 + 104 + 25 = 2065.7\ m\text{s} \\ Fi\_b[1] = 104 + 25 + 1870 + \dfrac{100}{1.5} = 2065.7\ m\text{s} \end{cases}$$

Since *Fi_a*[1] is equal to *Fi_b*[1], there is no queuing delay in the first IS (*q_Fi*[1] = 0), as it was expected. For the second IS, *Fi_a*[2] and *Fi_b*[2] can be computed using Eq. (7.18 or 7.19), as:

$$\begin{cases} Fi\_a[2] = \max\{2065.7,\ 2065.7\} + 22 + 25 = 2112.7\ m\text{s} \\ Fi\_b[2] = 104 + 25 + 746 + 25 + 1120 + \dfrac{100}{2} = 2070\ m\text{s} \end{cases}$$

Also in the second IS no queuing delay occurs, since *Fi_b*[2] is smaller than *Fi_a*[2]. Finally, for the third IS, *Fi_a*[3] and *Fi_b*[3] can be computed using Eq. (7.19):

$$\begin{cases} Fi\_a[3] = \max\{2112.7,\ 2070\} + 104 + 25 = 2241.7\ m\text{s} \\ Fi\_b[3] = 104 + 25 + 746 + 25 + 104 + 25 + 1870 + \dfrac{100}{1.5} = 2965.7\ m\text{s} \end{cases}$$

In this case, *Fi_b*[3] is greater than *Fi_a*[3], resulting in a worst-case queuing delay of *q_Fi*[3] = *Fi_b*[3] - *Fi_a*[3] = 2965.7 – 2241.7 = 724 μs. Finally, the sum the worst-case queuing delays in all ISs gives the worst-case total queuing delay that is equal to 724 μs (*Qfi* = 724 μs).

The worst-case total queuing Q (referenced with a (b), in Table 9.12) can be computed using (Eq. (7.22)), as follows (note that the worst-case queuing delays *Qdelta* and *Qfi* have the same value, only in this particular case):

$$Q = \max\{724,\ 724\} = 724\,\mu s$$

Computation of the maximum latency of the next (request or token) PDU

After receiving the token PDU from ES^5, ES^1 can issue a request PDU (from one of its message streams) or just pass the token to the next master ES (that is ES^5). In order to have a worst-case value for the system turnaround time, one must consider a worst-case scenario, where Eq. (7.30) must be maximised, for every possible length of a request DLL PDU, as expressed in Eq. (7.31).

The developed system planning software tool has been used, concluding that the maximum latencies achieved for the longest request PDU (L_{req} = 255, *ndp* = 4), i.e. using Eqs. (7.30) and (7.31) (referenced with (c), in Table 9.12):

$$MaxSum\_t_{sr} = 746 + 25 + 104 + 25 + 746 + 25 = 1671\,\mu s$$

Computation of t^{ndp}_{ID1}, C^{ndp}_{token} and C^{1}_{token}

The idle time (t^{ndp}_{ID1}) is the idle time of ES^1. Since this is a wired ES, its value is obtained just by dividing 375 (T^{wr}_{ID1}) by 1.5 (bit rate of WRD), resulting in 250 μs. The values for C^{ndp}_{token} (referenced with an (e), in Table 9.12) and C^{1}_{token} (referenced with an (f), in Table 9.12) can be obtained from Table 9.8, knowing that the first Communication Domain in the path is a WLD (C^{1}_{token}= 112 μs) and the last Communication Domain in the path is a WRD (C^{ndp}_{token}= 22 μs). Finally, it is possible to compute t_{st_token}, using Eq. (7.28) as (this result is referenced with a (g), in Table 9.12):

$$t_{st\_token} = 305 + 724 + 250 + 1671 + 22 - 112 = 2860\,\mu s$$

Figure 9.6 depicts all the parameters used for its computation (not at scale).

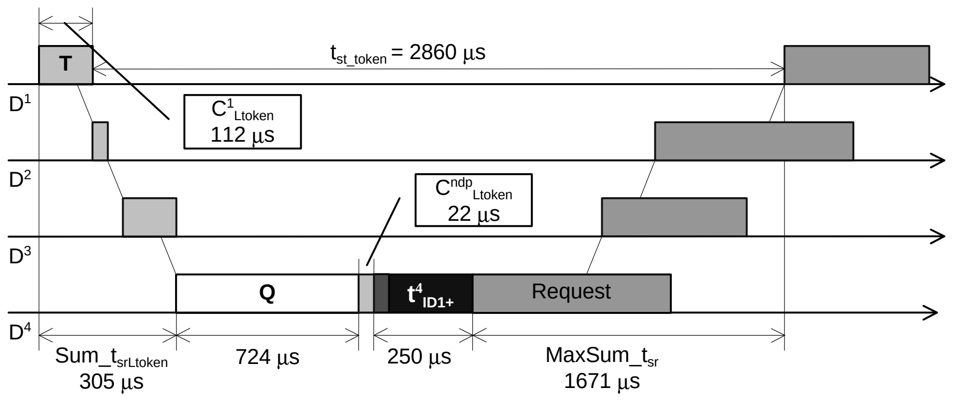


Figure 9.6: Parameters contributing to the computation of t_{st_token}

9.3. Case Study 2: a network scenario with inter-cell mobility

In this second case study, the example network presented for Case Study 1 is upgraded to support inter-cell mobility (Figure 9.7). Therefore, just the additional communication attributes related to this feature will be defined and the values for the additional parameters will be presented.

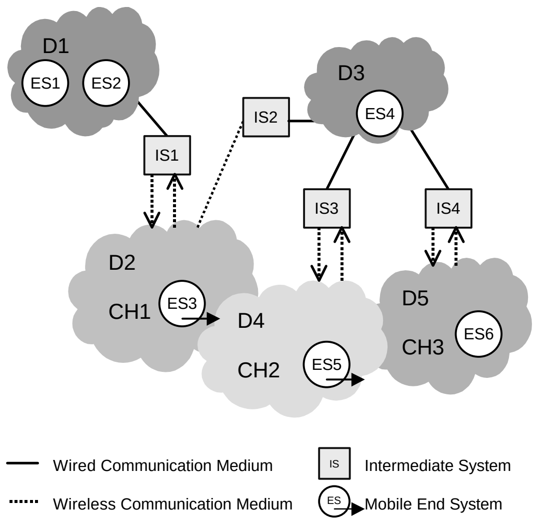


Figure 9.7: Network layout example considering inter-cell mobility

The differences, when comparing to Case Study 1 are:
- IS^1, IS^3 and IS^4 are now Structuring & Linking Intermediate Systems (SLIS);
- ES^1 supports the Mobility Management functionality (along with the normal functionality), i.e. is the Mobility Master (MobM);
- ES^3 and ES^5 are mobile ESs (able to join D^2, D^4 and D^5).

9.3.1. Communication Network model

Five additional parameters for the Communication Network must be considered. These are described in Table 9.13.

Table 9.13: Additional Communication Network parameters

Attribute	Value	Unit
L_{BT}	10	chars
nch	3	-
C_{beacon}	100	µs
t_{bgap}	25	µs
t_{sw}	100	µs

Communication Domains and Physical Media

The Communication Domains and Physical Media sets are equivalent to those of the first case study. Three Wireless Domains have changed from ad-hoc (AWLD) to structured (SWLD) Wireless Domains, in order to support inter-cell mobility. These differences to Table 9.2 are illustrated in Table 9.14.

Table 9.14: Models for the "new" Communication Domains

$D^2 = \left(SWLD, M^2, \{IS^1, IS^2\}, \{ES^3\}\right)$
$D^4 = \left(SWLD, M^2, \{IS^3\}, \{ES^5\}\right)$
$D^5 = \left(SWLD, M^2, \{IS^4\}, \{ES^6\}\right)$

End Systems and Intermediate Systems

The sets of End Systems (ESs) and Intermediate Systems (ISs) are also equivalent to those of the first case study. However, ES^1 is now also responsible for Mobility Management, being the Mobility Master (MobM), and ES^3 and ES^5 are now defined as mobile ESs (Table 9.15)

Table 9.15: Models for the "new" End Systems

$ES^1 = \left(WRES, M^1, MobM\right)$
$ES^3 = \left(MWLES, M^2, SLAVE\right)$
$ES^5 = \left(MWLES, M^2, MASTER\right)$

Finally, while IS^1, IS^3 and IS^4 were Linking Intermediate Systems (LISs) in the first case study, they are now Structuring & Linking Intermediate Systems (SLISs). Therefore, Table 9.6 is updated as described in Table 9.16.

Table 9.16: Models for the "new" Intermediate Systems

$IS^1 = \left(SLIS, \{M^1, M^2\}, -, -\right)$
$IS^3 = \left(SLIS, \{M^1, M^2\}, -, -\right)$
$IS^4 = \left(SLIS, \{M^1, M^2\}, -, -\right)$

9.3.2. Computation of T_{ID1} and T_{ID2}

The Idle Time parameters do not depend on the support of inter-cell mobility, as explained in Section 8.3.1. Therefore, the idle time parameters are similar to those of Case Study 1, except in what concerns the T_{ID2} parameter in the MobM, which must be computed in a different way.

9.3.3. Computation of t_{st} and T_{SL1}

The computation of T_{SL1} depends on the worst-case system turnaround times of all message transactions in the Communication Network. As it was already mentioned in

§8.3.2, for a particular message transaction, the fact that the initiator, the responder (or both) move between cells (or belong to a mobile WRD), can influence the worst-case system turnaround time for that message transaction. Therefore, in order to compute the worst-case system turnaround time and duration for a message stream (and T_{SL1}), every possible path between initiator and responder must be considered. This is reflected in the (text) file used as input data for the program that computes all network parameters, that is presented in Annex F.

Table 9.17 considers the different path possibilities as a result of inter-cell mobility. For instance, the responder of S^4 (ES^3) is mobile. Therefore, it can belong to D^2 (the case depicted in Figure 9.7), D^4 or D^5. As a consequence, the path between ES^1 and ES^3 changes, depending of the location of ES^3. If ES^3 is located in D^2, the path is $\{M^1, M^2\}$. If ES^3 moves to D^4, the path is $\{M^1, M^2, M^1, M^2\}$. If ES^3 joins D^5, the path is the same as the previous, i.e. $\{M^1, M^2, M^1, M^2\}$, so it is not worthwhile to consider it, since it will lead to the same results.

Table 9.17: System turnaround times and duration of message transactions

Message Stream	Initiator ES	Responder ES	L_{req}/L_{resp} (chars)	Path (M,…)	t_{stn} (µs)	Q (µs)	t_{st} (µs)	C_{ack} (µs)
S^4	ES^1	ES^3	255/6	1,2	200	0	200	2364
				1,2,1,2	**1276**	**0**	**1276**	**3440**
S^5	ES^1	ES^3	59/59	1,2	200	0	200	1315.3
				1,2,1,2	**693.3**	**590**	**1283.3**	**2398.7**
S^6	ES^1	ES^3	6/255	1,2	306	0	306	2470
				1,2,1,2	**1382**	**660.7**	**2042.7**	**4206.7**
S^{16}	ES^5	ES^2	255/6	2,1,2,1	2052	0	2052	4919.5
				2,1	**976**	**0**	**976**	**3843.5**
S^{17}	ES^5	ES^2	59/59	2,1,2,1	886.7	653.2	1539.9	3835.3
				2,1	**393.3**	**0**	**393.3**	**2688.8**
S^{18}	ES^5	ES^2	6/255	2,1,2,1	1946	723.9	2669.9	5537.3
				2,1	**870**	**0**	**870**	**3737.5**

Note that for S^4, the mobility of ES^3 makes the worst-case system turnaround time (t_{st}) rise from 200 µs to 1276 µs. The results for the message streams where the initiator is ES^5 must also take into account its mobility. Message streams S^{13}, S^{14} and S^{15} are not influenced by mobility, since independently of ES^5 being in D^2, D^4 or D^5, the *path* between ES^5 and ES^4 is always $\{M^2, M^1\}$. However, the case of message streams S^{16}, S^{17} and S^{18} is different. If ES^5 joins D^2, the *path* between ES^5 and ES^2 changes from $\{M^2, M^1, M^2, M^1\}$ (situation illustrated in Figure 9.7) to $\{M^2, M^1\}$. In this case, t_{st} does not change, since the alternative paths lead to lower worst-case system turnaround times.

In spite of the consideration of the additional possible paths imposed by the mobility of ES^3 and ES^5, the worst-case turnaround time of all message transactions remains the same (2669.9 µs). Therefore, the first component of the slot time parameter also remains equal to the first case study, i.e. t_{SL1} = 2669.9 µs. Oppositely, if ES^5 originally belonged to D^2, t_{SL1} would increase when considering its mobility.

It is also important to stress that the support of mobility always turns t_{SL1} greater or equal to the one without considering mobility.

9.3.4. Computation of T_{SL2} and T_{SL}

Since ES^5 is a master ES, its mobility feature also influences the computation of T_{SL2}. Mobility is taken into account by considering the additional paths for the token PDUs. Therefore, it must be taken into account that ES^5 can belong to D^4 (the case depicted in Figure 9.7), D^2 or D^5. Again, this is reflected in the (text) file used as input data for the program that computes all network parameters, which is presented in Annex F.

As an example, consider the token passing from ES^1 to ES^5. The path of the token PDU will change, depending of the location of ES^5. If ES^5 is located in D^4, the *path* is $\{M^1, M^2, M^1, M^2\}$. If ES^5 joins D^5, the *path* is the same as the previous, i.e. $\{M^1, M^2, M^1, M^2\}$, so it is not worthwhile to consider it, since it will lead to the same results. However, if ES^5 moves to D^2, the path changes to $\{M^1, M^2\}$.

Table 9.18 summarises some relevant parameters for the computation of the worst-case system turnaround time after a master ES passing the token to its successor. These parameters were computed using the software tools implementing the algorithms described in Annexes B, D and E.

Table 9.18: Parameters for the computation of T_{SL2}

Token passing	Path	$Sum\_t_{srtoken}$ (µs)	Q (µs)	t^{ndp}_{ID1} (µs)	$MaxSum\_t_{sr}$ (µs)	C^{ndp}_{token} (µs)	C^{1}_{token} (µs)	t_{st_token} (µs)
$ES^1 \rightarrow ES^5$	1,2,1,2	223	660.7	1623.5	1029	112	22	**3626.2**
	1,2	47	0	1623.5	129	112	22	1889.5
$ES^5 \rightarrow ES^1$	2,1,2,1	305	723.8	250	1671	22	112	2859.8
	2,1	129	0	250	771	22	112	1060

The second component of the Slot Time parameter – t_{SL2} – remains equal to the first case study, i.e. t_{SL2} = 3626.2 µs (bolded value in Table 9.18). Therefore, the same applies to the Slot Time parameter, i.e. $t_{SL} = max\{2669.8, 3626.2\} = 3626.2$ µs.

Note that t_{SL2} is not always greater than t_{SL1}. If the two master ES where located in the same Communication Domain (or at least nearer), t_{SL2} would be smaller and probably smaller than t_{SL1}. For instance, if master ES^5 was not mobile and belonged to D^2 (nearer ES^1), the second component of the Slot Time would be smaller ($t_{SL2} = 1889.5$ *ms*).

9.3.5. Computation of the Mobility Management parameters

Table 9.19 presents some relevant Mobility Management parameters for each SLIS in the Communication Network, computed using the previously referred software tool.

Table 9.19: Parameters for Mobility Management

SIS	Path	t_{btn} (µs)	Q (µs)	t_{bt} (µs)	t'_{bp} (µs)	n_b	t_{bp} (µs)	t_{mob} (µs)
IS^1	1,2	113.(6)	0	113.(6)	1711.(6)(c)	14(e)	1750(f)	1863.(6)(g)
IS^3	1,2,1,2	289.(6)(a)	660.(6)	950.(3)(b)	1535.(6)(d)	13(h)	1625(i)	2575(j)
IS^4	1,2,1,2	289.(6)	660.(6)	950.(3)	1535.(6)	13	1625	2575

IS^3 and IS^4 have equal values for all the parameters, since the *path* between the MobM and the correspondent SWLDs (SWLD2 and SWLD3, respectively) is equal.

Preliminary mobility management duration (t'_{mob})

The preliminary (worst-case) duration of the mobility management procedure (t'_{mob}) can be computed using Eq. (8.1), where the worst-case latency of the BT PDU from the MobM until a specific SWLD (t_{bt}) can be computed using Eqs. (8.2) and (8.3), as exemplified for the case of IS^3. The number of domains in the path (*ndp*) is 4, and the duration of the BT PDU is 73.3 µs in WRDs and 140 µs in WLDs (refer to Table 9.8):

$$t_{btn} = 22 + 25 + 104 + 25 + 22 + 25 + 140 - 73.3 = 289.(6)\ \mu s$$

This result (referenced with (a), in Table 9.19). Using Eq. (8.2) and having computed the value of *Q* (not described), the maximum latency for the BT PDU (t_{bt}) is:

$$t_{bt} = 660.(6) + 289.(6) = 950.(3)\ \mu s$$

This value is referenced with a (b), in Table 9.19. The maximum duration of the handoff procedure (t_{ho}) can be computed using Eq. (8.12), considering *nch* = 3:

$$t_{ho} = 5 \cdot 100 + 3 \cdot 25 + 3 \cdot 100 = 875\ \mu s$$

Therefore, t'_{mob} results in (Eq. 8.1):

$$t'_{mob} = 950.3 + 875 = 1825.(3)\ \mu s$$

Preliminary duration of the beacon period (t'_{bp})

The preliminary (worst-case) duration of the beacon period, for IS^1 and IS^3 (IS^4) can be computed using Eq. (8.15) as follows (referenced with (c) and (d), in Table 9.19):

$$t'_{bp}(IS^1) = 1825.(3) - 113.(6) = 1711.(6)\ \mu s$$

$$t'_{bp}(IS^3) = 1825.(3) - 289.(6) = 1535.(6)\ \mu s$$

Number of beacons for IS^1

The number of beacons for IS^1 ($n_b(IS^1)$) can then be computed using Eq. (8.16) as follows (this result is referenced with an (e), in Table 9.19):

$$n_b(IS^1) = \left\lceil \frac{1711.(6)}{25 + 100} \right\rceil = 14$$

Re-computing the beacon period duration for IS^1 using Eq. (8.17), results in (this result is referenced with an (e), in Table 9.19):

$$t_{bp}(IS^1) = 14 \cdot 125 = 1750\ \mu s$$

Finally, the worst-case mobility management duration due to IS^1 can be computed using Eq. (8.18), as follows (referenced with a (g), in Table 9.19).

$$t_{mob}(IS^1) = 113.(6) + 1750 = 1863.(6)\ \mu s$$

Number of beacons for IS^3 and IS^4

Similarly, the number of beacons for IS^3 (IS^4) can also be computed using Eq. (8.16) as follows (this result is referenced with a (h), in Table 9.19):

$$n_b(IS^3) = \left\lceil \frac{1535.(6)}{25+100} \right\rceil = 13$$

Re-computing the beacon period duration for IS^3 (IS^4) also using Eq. (8.17) (this result is referenced with an (i), in Table 9.19):

$$t_{bp}(IS^3) = 13 \cdot 125 = 1625\,\mu s$$

The worst-case mobility management duration due to IS^3 (IS^4) is (referenced with a (j), in Table 9.19):

$$t_{mob}(IS^3) = 950.(3) + 1625 = 2575.(3)\,\mu s$$

After having computed the mobility management duration for all SWLD (and the number of beacons for all the corresponding SIS/SLIS), the maximum duration of the mobility management procedure is (Eq. 8.19):

$$t_{mob} = \max\{1863.(6), 2575.(3), 2575.(3)\} = 2575.(3)\,\mu s$$

Finally, the idle time parameter T_{ID2} for the MobM should be set to (Eq. (8.20)):

$$T_{ID2} = \lceil 2575.(3) \cdot 1.5 \rceil = 3863\,\text{bits}$$

Note that bit rate r_{MobM} is equal to 1.5 Mbit/s, since the MobM belongs to a WRD.

9.4. Mobility Management parameters as a function of the location of the MobM and of its type

This additional numerical analysis reinforces what was previously stated in Section 8.4.6, namely concerning the location of the Mobility Master.

9.4.1. Mobility Management parameters for the MobM in D^3

If ES^1 is relocated in D^3, there is the need to re-compute t_{SL1} and t_{SL2}, due to the change in the paths of the message streams of ES^1 and in the paths of the token to and from ES^1. Nonetheless, only the Mobility Management parameters will be addressed. Using the already referred software tool, the results presented in Table 9.20 are obtained.

Table 9.20: Parameters for Mobility Management

SIS	Path	t_{btn} (μs)	Q (μs)	t_{bt} (μs)	n_b	t_{bp} (μs)
IS^1	1,2,2	242.7	0	242.7	7	875
IS^3	1,2	113.7	0	113.7	9	1125
IS^4	1,2	113.7	0	113.7	9	1125

Note that the *path* between the MobM and an ES in D^2 is $\{M^1, M^2, M^2\}$, since after being relayed by IS^2 (a LIS), the BT PDU must be relayed by IS^1 (an SLIS), in order to be received by an ES in D^2.

Since the paths between the MobM and the SWLDs are generally shorter (less ISs) than for the original location of ES^1, latencies of the BT PDU are smaller. For instance, the worst-case queuing delay affecting the BT PDU (caused by a preceding token PDU - Q_D) is zero for all SWLDs. As a consequence, the number of beacons each SLIS must issue is also smaller. The idle time parameter T_{ID2} in the mobility master should be set to a minimum value of *1858 bits* (less than a half of the idle time for ES^1 in D^1).

9.4.2. Mobility Management parameters for exclusive MobM in D^1

If a dedicated master ES is used for the mobility management, an additional master ES must be added to the Communication Network of Case Study 2. Obviously, this additional master ES will influence the computation of t_{SL2}, but only the Mobility Management parameters will be addressed here. The results (again, obtained using the previously mentioned software tool) are presented in Table 9.21.

Table 9.21: Parameters for Mobility Management

SIS	Path	t_{btn} (μs)	Q (μs)	t_{bt} (μs)	n_b	t_{bp} (μs)
IS^1	1,2	113.7	0	113.7	9	1125
IS^3	1,2,1,2	289.7	**0**	289.7	7	875
IS^4	1,2,1,2	289.7	**0**	289.7	7	875

The fact that there are no additional message streams in the MobM influences the worst-case queuing delay affecting the BT PDU, which are now zero (bolded values in Table 9.21). This has a great impact on the number of beacons (that is much smaller than for MobM embodied in ES^1) and on the idle time parameter T_{ID2} in the mobility master (it decreases to 1858 bits, less than a half of the idle time for MobM integrated in ES^1).

As a conclusion, both the location of the MobM and whether it is exclusively dedicated to Mobility Management or not, must be carefully analysed, in order to have an optimised system performance. The lower the idle time the MobM must insert (t_{ID2}), the smaller the time spent in the Mobility Management, resulting in a better data throughput.

9.5. Using the developed tools to analyse the characteristics of the architecture

Up to now, this chapter has focused on the application of the proposed methodologies to two different case studies. In this section, the analysis will be extended to cover characteristics inherent to the overall architecture. To this purpose, the software tool implementing the algorithms described in Annexes B, D and E was used.

9.5.1. Idle Time parameters

Idle Time parameters as a function of the bit rate

The variation of the bit rate of the wired Physical Media has an important impact on the evaluation of the Idle Time parameters.

PROFIBUS-DP supports 9.6, 19.2, 93.75, 187.5 and 500 kbit/s and 1.5, 3, 6 and 12 Mbit/s bit rates. The attributes of the Communication Network are the ones assumed for the case studies (defined in §9.2.1). The bit rate of the WLD is assumed to be 2 Mbit/s (as in both case studies).

Table 9.22 depicts the inserted idle time values, considering several bit rates for the WRD (the inserted idle times are presented in time units rather than in bit times, since there are different bit rates involved). The lowest two bit rates (9.6 and 19.2 kbit/s) have not been considered, since they lead to very high inserted idle times (these bit rates are too low, when compared to the bit rate for the WLDs - 2 Mbit/s). In the table, the values are rounded to the nearest integer, for the sake of simplicity.

Table 9.22: Idle Time parameters for different WRD bit rates

WRD bit rate (Mbit/s)	t^{wr}_{ID1+} (μs)	t^{wr}_{ID2+} (μs)	t^{wl}_{ID1+} (μs)	t^{wl}_{ID2+} (μs)
0.09375	0	0	59673	29817
0.1875	0	0	28687	14323
0.5	0	0	9320	4640
1.5	183	63	1573	766
3	427	202	0	0
6	1378	686	0	0
12	1854	928	0	0

For low bit rates (smaller or equal to 0.5 Mbit/s), there is no need to for the WRES to insert idle time, since WRDs are much slower than WLDs. There is no risk of a WRES to cause traffic congestion (increasing queuing) in an IS, since IS are able to relay PDUs from WRD to WLD without any problem. Oppositely, WLES are forced to insert very high inactivity times (almost 60 ms, in one case), since wireless PhL PDUs are much shorter than wired PhL PDUs.

When the bit rate (of WRDs) grows beyond 1.5 Mbit/s, the idle time that must be inserted by WRESs increases. However, these inserted idle times are much smaller (below 2 ms) that the ones of WLES for low wired bit rates. The main reason for this is that the wireless PhL PDU has a fixed overhead per PhL PDU (200 bits of preamble, start frame delimiter and header), wired PhL PDUs have a fixed overhead per DLL char (start, parity and stop bits), which turns out to be significant for high DLL PDU lengths. In turn, the inserted idle time in WLES is null, since it is impossible for WLES to cause traffic congestion in the ISs (even for the smallest PDU length).

Idle Time parameters as a function of the maximum PDU length

It is going to be shown next how the variation of the maximum (request/response) PDU length influences the Idle Time parameters.

Due to the factor ‘$(d + k)/r$’ (8/2 < 11/1.5), the idle time parameters (T_{ID1} and T_{ID2}) of wired ESs do not depend on the maximum PDU length, but on the minimum PDU length. Therefore, only the idle time parameters (T_{ID1} and T_{ID2}) of wireless ESs will be addressed.

Figure 9.8 depicts a chart of the idle time parameters of wireless ESs as a function of the maximum length of request/response PDUs (it is assumed that $L^{max}_{req} = L^{max}_{resp}$). For $L^{max}_{req} = L^{max}_{resp}$ = 19 chars, T^{wl}_{ID1} = 100 bits and T^{wl}_{ID2} = 100 bits, while for $L^{max}_{req} = L^{max}_{resp}$ = 255 chars, T^{wl}_{ID1} = 3247 bits and T^{wl}_{ID2} = 1634 bits. Both idle time parameters increase linearly with the maximum DLL PDU length.

Obviously, the maximum length of DLL PDUs should be kept as low as possible, in order to minimise the value of the Idle Time parameters. The lowest the value of the Idle Time parameters, the lowest inactivity periods, leading to lower duration of message transactions, system turnaround times and Slot Time, and higher responsiveness. For instance, if the maximum length of the DLL PDUs, for all message streams in the Communication Network, is 59 chars (50 data octets), then the idle time parameters can be computed by considering this (59) as the maximum length PDU (5 times smaller than for 255 chars maximum length).

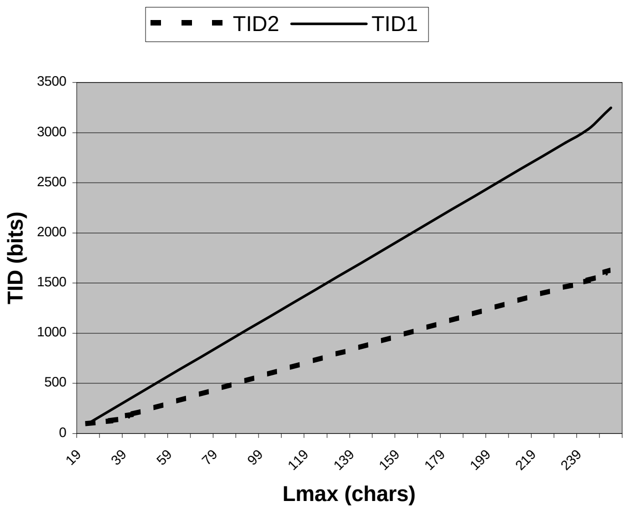


Figure 9.8: Variation of T_{ID1} and T_{ID2} with L^{max}

9.5.2. Slot Time parameter

Slot Time parameters as a function of the bit rate of WRDs

The impact of the variation of the bit rate of WRDs in the Slot Time parameter is going to be analysed next, using the Communication Network scenario presented for Case

Study 2 as an example. Additionally, different bit rates for the wired Physical Medium (assuming always a 2 Mbit/s bit rate for WLDs) will be considered. The results are presented in Table 9.23.

Table 9.23: Slot Time components for different WRD bit rates

WRD bit rate (Mbit/s)	t_{SL1} (µs)	t_{SL2} (µs)
0.09375	87480	118946
0.1875	42248	57506
0.5	13978	19106
1.5	2670	3626
3	879	1049
6	2282	2901
12	2983	3831

At a first glance, it may seem awkward that t_{SL1} for 3 Mbit/s is smaller than t_{SL1} for 6 Mbit/s, and the latter is smaller than t_{SL1} for 12 Mbit/s. This is justified by the following example. If an initiator belonging to a WRD issues a maximum request PDU, its duration for 3 Mbit/s (935 µs) is twice the duration for 6 Mbit/s (468 µs). However, the start relaying instant from a WRD to a WLD is almost equal (33/3 for 3 Mbit/s and 33/6 for 6 Mbit/s). Therefore, since the request PDU for 6 Mbit/s ends earlier than for 3 Mbit/s, the system turnaround time will be greater, as illustrated in Figure 9.9.

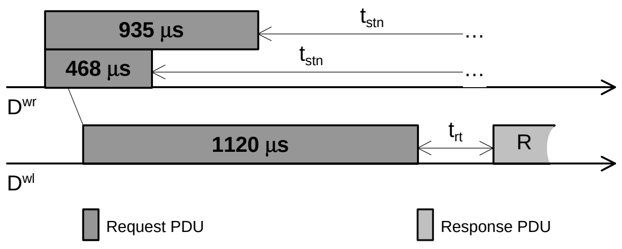


Figure 9.9: The impact of different bit rates on t_{st}

The same phenomenon happens with the worst-case queuing delay, i.e. Q is greater for increasing bit rates (of the wired Physical Medium), above a certain threshold. Therefore, both t_{SL1} and t_{SL2} increase with the increase in the referred bit rate.

Slot Time parameters as a function of the maximum PDU length

The impact of the variation of the maximum PDU length on t_{SL1}, considering only one message stream in the Communication Network, will be analysed next. The chosen message stream is S^{11} of Case Study 1 (ES^1 as initiator, ES^6 as responder and $L_{req} = L_{resp} = 59$ chars). For this purpose, maximum DLL PDU lengths ($L^{max}_{req} = L^{max}_{resp} = L^{max}$) of 255, 239, 229, …, 59 chars are considered. For the computation of t_{SL2}, masters ES^1 and ES^5 (as in the case studies) are assumed. Mobility is not considered.

Table 9.24: Slot Time components for different maximum PDU lengths

L^{max} (chars)	Q_{req} (μs)	t_{SL1} (μs)	t_{SL2} (μs)
255	590	1283	3626
239	537	1230	3413
229	503	1197	3279
…	…	…	…
89	37	730	1413
79	3	697	1279
69	0	693	1146
59	0	693	1013

Note that as the maximum length (L^{max}) decreases, so decrease t_{SL1} and t_{SL2}. The former decreases proportionally to the decrease in the worst-case total queuing delay affecting the request PDU (Q_{req}). The slot time component t_{SL2} decreases at a higher rate, since it does not depend only on the worst-case queuing delays affecting the token PDUs, but also on the idle time t_{ID1} of the successor, which both decrease with the decrease of L^{max}.

9.6. Summary

This chapter addressed the practical application of the models and methodologies proposed in Chapters 5, 6, 7 and 8. All the numerical results were obtained by using a system planning software application developed in the context of this thesis, which is based on the algorithms presented in Annexes B, D and E.

The models for all the objects considered in the Communication Network scenarios, namely for the Communication Domains, Physical Media, End Systems, Intermediate Systems and Message Streams have been defined and all the relevant networks parameters were computed, for the particular case studies.

Chapter 10

Conclusions

This chapter reviews the research objectives of this thesis and summarises its major results, highlighting how the research contributions fulfilled the original research objectives. Moreover, some guidelines for future research work in the area of hybrid wired/wireless fieldbus networks are provided.

10.1. Review of the research objectives

The communication infrastructure of current Distributed Computer-Controlled Systems (DCCS) is usually based on fieldbus networks, since they provide adequate levels of performance, dependability, timeliness, maintainability and cost. Nevertheless, cabling starts to be an obstacle for an increasing number of industrial automation applications, which impose or benefit from the use of mobile devices.

Within this context, there is a trend to extend fieldbus systems with wireless capabilities. Considering the evolution of wireless local area network (WLAN) technologies, which are targeted to office applications, it would seem reasonable to use these standardised systems (e.g. IEEE 802.11b) in industrial automation scenarios as well. However, constraints such as insufficient performance, low level of dependability and the inappropriateness of wireless MAC protocols to ensure the real-time behaviour of the wireless network make them inadequate for industrial DCCS applications.

Current (wired) fieldbus networks provide high levels of performance and dependability, as well as real-time behaviour. Therefore, wireless extensions must not disrupt these characteristics of fieldbus networks.

The main research objectives of this thesis were the design of a hybrid wired/wireless communication architecture based on a standard fieldbus protocol, and the proposal and discussion of the appropriate mechanisms and approaches to support and guarantee real-time communications with such an architecture. The hypothesis was that such an architecture was possible. As outlined next, the research contributions of this thesis fulfilled the objectives and enabled the confirmation of the hypothesis.

10.2. Main research contributions

This thesis provides the following contributions to the development of hybrid wired/wireless fieldbus networks.

10.2.1. Architecture for a hybrid wired/wireless fieldbus network

This thesis addresses the extension of a traditional (wired) fieldbus network to support wireless and mobile nodes. This hybrid wired/wireless network could be based on different fieldbus systems, but PROFIBUS-DP has been elected as the federating communication system for the hybrid communication network, for which rationale for the choice was given in Chapter 3. Nonetheless, most of the approaches could be potentially used in other standard fieldbus networks.

Then, in Chapter 4, some design alternatives were analysed and compared, justifying the choice for an approach where the Wired and Wireless Domains are interconnected through Intermediate Systems (ISs) behaving as repeaters. In addition to this, several design rules that govern the proposed approach were presented and an innovative mechanism that is able to support inter-cell mobility with the desired requirements was described. One of the advantages of this architecture is that it requires no changes to the PROFIBUS core protocols.

In Chapter 5, analytical models for such a Communication Network were proposed, to enable the analysis carried out in Chapters 6, 7 and 8. These models provided parameters and behaviour for a number of different components, namely for Communication Domains, Physical Media, Intermediate Systems (ISs) and End Systems (ESs). Importantly, the model for the Physical Media included a generic format for Physical Layer Protocol Data Units (PhL PDUs), and the model for the ISs supported both cut-through and store&forward behaviours. These aspects were of paramount importance in the scope of this thesis, since they showed to have a strong impact on very important time parameters.

10.2.2. Traffic adaptation through insertion of additional idle time

The heterogeneity in bit rates and in PhL PDU formats in such a broadcast network imposes the consideration of some kind of traffic adaptation scheme, since traffic congestion may occur (in the ISs). The problem originated by this heterogeneity in the physical layers is that message response times are affected by increasing delays due to queuing in the ISs. To our best knowledge, there is no related work focusing on this problem. Therefore, in order to achieve reduced and bounded message response times, a specific mechanism to eliminate these queuing delays was proposed in Chapter 6.

The proposed mechanism relies on an appropriate setting of the PROFIBUS Idle Time parameters (T_{ID1} and T_{ID2}). The Idle Time is an inactivity time that must be respected by master ESs before transmitting request or token PDUs. This reduces the number of transactions per time unit, when the responder is not located in the same Communication Domain as the initiator. However, the advantage of the proposed methodology is enormous, as it leads to a better responsiveness to failures and to bounded and smaller worst-case message response times, and does not impose any changes to the PROFIBUS protocol. Moreover, it enables the computation of the worst-case duration for any message transaction and of the PROFIBUS Slot Time parameter (as detailed in Chapter 7).

It is also important to note that this methodology can be applied to any type of broadcast network composed by heterogeneous transmission media.

10.2.3. Computation of the duration of message transactions and of the Slot Time parameter

The message's response time in a PROFIBUS-based Communication Network is mainly dependent on the medium access delay (contention due to other messages in the queue and due to other stations holding the token) and on the duration of the message transaction. Such duration includes both the duration of the request/response PDUs and the system turnaround time associated with that transaction, that is, the time interval between the end of the request transmission and the beginning of the response reception.

Since both the initiator and the responder ESs may belong to different Communication Domains, request and response PDUs may have to be relayed by one or more ISs before reaching the destination. Therefore, a specific analysis was required to evaluate such additional latencies introduced by the hybrid nature of the network .

Chapter 7 presented a methodology to evaluate the worst-case system turnaround time (t_{st}) and worst-case duration (C_{ack}) of message transactions, enabling the setting of the PROFIBUS Slot Time parameter (T_{SL}). This parameter defines the timeout before which a response/acknowledgement must arrive, and it is also used for the token recovery mechanism. Therefore, T_{SL} assumes a particular importance, for such a hybrid network. On one hand, T_{SL} must be set large enough to cope with the extra latencies introduced by the ISs. On the other hand, T_{SL} must be set as small as possible such as the system responsiveness to failures does not decrease dramatically and worst-case message response times are as small as possible.

10.2.4. Timing analysis considering inter-cell mobility

Chapter 8 presents a timing analysis of the mobility management mechanism that was introduced in Chapter 4. This mechanism uses native PROFIBUS features and provides a seamless handoff for mobile master and slave ESs and also for Mobile Linking ISs (MLISs). Basically, a mobility manager (MobM) is responsible for starting a well-defined mobility management period, during which every mobile system will be able to perform the handoff procedure. One of the pros of this mechanism is its associated timing determinism, since the mobility management duration can be determined *a priori*.

In order for this mechanism to be compatible with the characteristics of PROFIBUS, after the Mobility Master (MobM) triggers the mobility management mechanism, it must insert an adequate idle time (T_{ID2}) corresponding to the duration of the mobility management period, before issuing another transaction or passing the token. The duration of this mobility management period depends on the number of beacons (n_b) that each Structuring Intermediate System (SIS) or Structuring & Linking Intermediate System (SLIS) must transmit, for radio channel assessment purposes. A methodology to compute both the worst-case duration of the mobility management period and the proper number of beacons to be transmitted by each SIS/SLIS was proposed in Chapter 8.

The impact of the inter-cell mobility in the Idle Time and Slot Time parameters is also addressed in Chapter 8. While the Idle Time parameters only depend on the Physical Media existent in the Communication Network, the Slot Time is influenced by inter-cell mobility, due to changes in the path (number and type of physical media) between initiator and responder or between a master and its successor.

10.3. Future work

Although the architectural models and mechanisms proposed in this thesis guarantee the real-time behaviour of the Communication Network, some additional work and improvements can still be made, as outlined next.

The numerical results presented in Chapter 9 show the impact of the variation of the bit rate of Wired Domains in the Idle Time and Slot time parameters. As it was already mentioned, these parameters should be kept as small as possible, in order to increase data throughput. If wired and wireless bit rates are rather different, the inserted idle times and system turnaround times increase significantly. For the scenarios proposed in Chapter 9, the optimal situation seems to be for the wired bit rate slightly above the wireless bit rate. Nevertheless, a more detailed analysis could be devoted to the effect of considering other types of PhL PDU formats and bit rates in relevant network parameters.

Since the insertion of additional idle time reduces the data throughput (data/idle ratio) of the network, it would be interesting to analyse and eventually measure this throughput for different scenarios. Another issue to be further investigated is the simultaneous use of cut-through and store&forward ISs in a Communication Network. For instance, it could be mandatory for the structuring types of ISs (SISs/SLISs) to relay PDUs (from wireless to wireless) in a store&forward way, while Linking ISs (LISs) could behave as cut-through repeaters. This would require a re-formulation of the analysis and methodologies for the computation of the Idle Time and Slot Time parameters.

To compute the worst-case queuing delay (Q) affecting a request PDU, it was considered that this PDU is preceded by an infinite sequence of maximum length PDUs. Future work could focus on trying to reduce this pessimism, eventually by considering possible sequences of message transactions and seeing the effect on a request PDU. Apparently, the number of PDUs that the ISs must be able to store increases with the number of ISs in the Communication Network. An interesting issue to be further investigated would be analysis of buffer requirements, both for cut-through and store&forward ISs.

Most real-time systems are also safety-critical systems, demanding a high level of dependability. Therefore, another extension to this work would be to combine the proposed timing analysis with some dependability analysis. How the network behaves in the presence of faults, e.g., the impact of a mobile ES getting out of range when transmitting or receiving. Concerning the mobility management mechanism, which mechanisms could be introduced to support some fault-tolerance level and how these mechanisms would affect the real-time behaviour of the Communication Network.

The addressed communication architecture is based on ISs acting as repeaters. An alternative approach would be an architecture based on bridges/routers. Although this approach requires some more complex mechanisms (e.g. mobility management) and changes to the PROFIBUS (MAC) protocol, its potential could be rewarding in terms of dependability and timeliness (Ferreira et al., 2002).

Finally, it would be also interesting to consider other state-of-the-art fieldbus protocols besides PROFIBUS (e.g. WorldFIP, Foundation Fieldbus) as possible federating communication systems and analyse different alternatives for the hybrid architecture.

References

AEG/Schneider Automation Inc. (1996). Modicon Modbus Plus Network Planning and Installation Guide. 890 USE 100 00, Version 3.0, April 1996.

Agrawal, G., Chen, B., Zhao, W. and Davari, S. (1994). Guaranteeing Synchronous Message Deadline with the Timed Token Medium Access Control Protocol. In IEEE Transactions on Computers, Vol. 43, No. 3, pp. 327-339.

Alves M., Bangemann T., Batista B., Breithaupt R., Ferreira L., Haehniche J., Hammer G., Heidel R., Kalivas G., Kalogeras A., Kapsalis V., Koubias S., Koulamas C., Kutschenreuter M., Monforte S., Pacheco F., Roether K., Speckmeier P., Tovar E., Vasques F. (2000d). General System Architecture of the RFieldbus. Deliverable D1.3. RFieldbus project IST-1999-11316.

Alves M., Batista B., Ferreira L., Hammer G., Heidel R., Kalivas G., Kalogeras A., Kapsalis V., Karavasilis C., Koubias S., Koukourgiannis A. (2000c). Requirements for the RFieldbus System. Deliverable D1.1. RFieldbus project IST-1999-11316, May 2000.

Alves M., Tovar E., Fohler G., Buttazzo G. (2000b). CIDER – Envisaging a COTS Communication Infrastructure for Evolutionary Dependable Real-Time Systems. WIP Session of the 12th Euromicro Conference on Real-Time Systems, Stochholm, Sweden.

Alves M., Tovar E., Vasques F. (2000a). Ethernet goes real-time: a survey on research and technological developments. Technical Report HURRAY-TR-2k01, January 2000.

Alves M., Tovar E., Vasques F. (2001a). On the Adaptation of Broadcast Transactions in Token-Passing Fieldbus Networks with Heterogeneous Transmission Media. In Proc. of the 4th IFAC Conference on Fieldbus Systems and their Applications (FeT'01), Nancy, France, pp. 278-284.

Alves M., Tovar E., Vasques F. (2001b). Evaluating the Duration of Message Transactions in Broadcast Wired/Wireless Fieldbus Networks. In Proc. of the 27th Annual Conference of the IEEE Industrial Electronics Society (IECON'01), Denver, EUA, pp. 243-248.

Alves M., Tovar E., Vasques F., Hammer G., Röther K. (2002). Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-based Networks. Proc. of the 14th Euromicro Conference on Real-Time Systems – ECRTS'02, Vienna, Austria, pp. 142-150.

Beeston J. (2001). WorldFIP in the Real-Time World. World Bus Journal, www.isa.org.

Benito M., Fuertes J., Kahoraho E., Arzoz N. (1999). Performance Evaluation of Four Fieldbuses. Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'99), Barcelona, Spain, pp. 881-890.

Bilstrup U., Wiberg P.-A. (2000). Bluetooth in Industrial Environment. Proc. of the 3rd IEEE International Workshop on Factory Communication Systems (WFCS'00), Porto, Portugal, pp. 239-246.

Blanes J., Sempere P., Victor M. (2001). Machine Vision in PROFIBUS Networks. In Proc. of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Luan les Pins, France, pp. 367-375.

Bluetooth Specification (1999). Specification of the Bluetooth System. Version 1.0 B. Specification Volume 1, Document No.: 1.C.47/1.0 B.

BSI (2000). BSI Guide to the Evaluation of Fieldbus Protocols – Selecting the best Fieldbus for your Application. BSI Standards, London, UK.

Carvalho A., Portugal P. (2001). On Dependability Evaluation of Fieldbus Networks: a transient fault analysis. In Proc. of the 4th IFAC International Conference on Fieldbus Systems and their Applications (FeT'2001), Nancy, France, pp.45-52.

Castelpietra P., Song Y-Q., Simonot-Lion F., O. Cayrol (2000). Performance Evaluation of a Multiple Networked in-Vehicle Embedded Architecture. Proc. of the 3rd IEEE International Workshop on Factory Communication Systems (WFCS'00), Porto, Portugal, pp. 187-194.

Cavalieri S., Di Stefano A., Mirabella O. (1994). Exploiting FDDI Communication Features to Connect FieldBuses in Process Control Environment. Proc of the IEEE Conference on Local Computer Networks – LCN'94, Minneapolis, USA.

Cavalieri S., Panno D. (1997). On the Integration of Fieldbus Traffic within IEEE 802.11 Wireless LAN. Proc. of the 2nd IEEE International Workshop on Factory Communication Systems, Barcelona, Spain, pp. 131-138.

CCE-CNMA (1994). EFACEC Detailed Pilot Specification. Deliverable 11 of the ESPRIT Project 7096 – CIME Computing Environment Integrating a Communication Network for Manufacturing Applications (CCE-CNMA).

Cena G., Demartini C., Valenzano A. (1997). On the Performance of Two Popular Fieldbuses. In Proc. of the 2nd IEEE Int. Workshop on Factory Communication Systems – WFCS'97, Barcelona, Spain, pp. 177-186.

Chávez M., Thomesse J.-P. (2000). Fieldbuses and Real-Time MAC protocols. In Proc of the 4th IFAC Intern. Symposium (SISICA'00), Buenos Aires, Argentina, pp. 51-56.

CNMA (1991). EFACEC Pilot Description. Deliverable of the ESPRIT Project 5104 - "Communication Networks for Manufacturing Applications (CNMA).

Coad P., Yourdon E. (1991). Object-Oriented Analysis. Second Edition. Yourdon Press, Prentice Hall, ISBN 0-13-629981-4, 1991.

Decotignie J.-D. (2001). A Perspective on Ethernet TCP/IP as a Fieldbus. In Proc. of the 4th IFAC International Conference on Fieldbus Systems and their Applications (FeT'2001), Nancy, France, pp.138-143.

Decotignie J.-D., Dallemagne, P., El-Hoiydi A. (2001). Architectures for the Interconnection of Wireless and Wireline Fieldbusses. In Proc. of the 4th Int. Conf. on Fieldbus Systems and their Applications (FET'01), Nancy, France, pp. 285-290.

Dietrich D., Sauter T. (2000). Evolution potentials for fieldbus systems. 3rd IEEE International Workshop on Factory Communication Systems, Porto, pp. 343-350.

El-Hoiydi A., Dallemagne P. (2000). Influence of Roaming on Real-Time Traffic in Wireless Networks. WIP Session of the 3rd IEEE International Workshop on Factory Communication Systems, Porto, Portugal.

El-Hoiydi A., Decotignie J.-D. (2001). Soft Deadline Bounds fo Two-Way Transactions in Bluetooth Piconets under co-channel Interference. Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Luan les Pins, France, pp. 143-150.

EN 300175 (2001). European Standard (Telecommunications series). Digital Enhanced Cordless Telecommunications (DECT). Common Interface (CI). Parts 1-9, Version V1.5.1.

EN 300652 (1998). Broadband Radio Access Networks (BRAN). High Performance Radio Local Area Network (HIPERLAN) Type 1 – Functional Specification. European Standard (Telecommunication Series), Version v1.2.1. Available (free of charge) at http://www.etsi.org

EN 50170 (1996). General Purpose Field Communication System. Volume 1 – P-NET, Volume 2 - PROFIBUS, Volume 3 – WorldFIP, European Norm.

EN 50254. (1996). High Efficiency Communications Subsystems for Small Data Packages. CLC TC/65CX, CENELEC EN 50254 Project.

ETSI (2000a). UMTS – Physical Layer. ETSI TS 125 201. V3.0.2.

ETSI (2000b). Broadband Radio Access Networks (BRAN). High Performance Radio Local Area Network (HIPERLAN) Type 2 – System Overview. TR 101 683 V1.1.1, http://www.etsi.org.

Ferreira L, Tovar E., Machado S. (2001). Scheduling IP Traffic in Multimedia Enabled PROFIBUS Networks. Proc. of the 8th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Luan les Pins, France, pp. 169-176.

Ferreira L., Alves M., Tovar E. (2002). Hybrid Wired/Wireless PROFIBUS Networks Supported by Bridges/Routers. Proc. of the 4th International Workshop on Factory Communication Systems (WFCS'02), Mälardalen University, Västerås, Sweden, pp. 193-202.

Fredriksson L.-B. (1999). Bluetooth in Automotive Applications. Proc. of the Bluetooth’99 Conference, London, UK.

Grow, R. (1982). A Timed Token Protocol for Local Area Networks. In Proc. of Electro’82, Token Access Protocols, Paper 17/3.

Hong S., Kim K. (1997). Implementation and Performance Evaluation of PROFIBUS in the Automation Systems. In Proc. of the 2nd IEEE Int. Workshop on Factory Communication Systems – WFCS’97, Barcelona, Spain, pp. 187-192.

Huselbos B. (2001). Enter Ethernet, Exit Fieldbus?. The online Industrial Ethernet Book, 2001.

Hutter S., Steiner R., (1999), “Connecting a P-NET Fieldbus System to Remote Operators via Internet”, Technical Description, ICT, Technical University of Vienna, V1.0.

IEEE 802.11 (1997). Wireless LAN Medium Access Control and Physical Layer Specification – 802.11. IEEE standard board, USA.

IEEE 802.11a (1999). Wireless LAN Medium Access Control and Physical Layer Specification – 802.11a. IEEE standard board, USA.

IEEE 802.11b (1999). Wireless LAN Medium Access Control and Physical Layer Specification – 802.11b. IEEE standard board, USA.

IEEE 802.1D (1998). Information technology--Telecommunications and information exchange between systems--Local and metropolitan area networks - Common Specification - Media access control (MAC) bridges. This is a revision of ISO/IEC 10038: 1993, 802.1j-1992 and 802.6k-1992. It incorporates IEEE Std 802.11c-1998, P802.1p and P802.12e.

IEEE 802.1Q (1998). IEEE Standard for Local and Metropolitan Area Networks: Virtual Bridge Local Area Networks.

IEEE 802.3 (1998) Edition, 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, This edition includes all contents of the 8802-3:1996 Edition, plus IEEE Std 802.3aa-1998, IEEE Std 802.3r-1996, IEEE Std 802.3u-1995, IEEE Std 802.3x&y-1997, and IEEE802.3z-1998.

ISO 11898 (1993). Road Vehicle - Interchange of Digital Information - Controller Area Network (CAN) for High-Speed Communication. ISO.

Jacobson V. (1988). Congestion Avoidance and Control. In Proc. of the SIGCOMM’88, Standforf, CA, USA.

Jain R. (1990). Congestion Control in Computer Networks: Issues and Trends. In IEEE Network Magazine, May 1990, pp. 24-30.

Jasperneite, J., Neumann, P. (2001). Switched Ethernet for Factory Communication. Proceedings of Emerging Technologies and Factory Automation, pp 205,212.

Knizak M., Kunes M., Manninger M., Sauter T. (1997), “Applying Internet Management Standards to Fieldbus Systems”, 2nd IEEE International Workshop on Factory Communication Systems, Barcelona, Spain, pp. 309-315.

Koulamas C., Lekkas A., Kalivas G., Koubias S., Roether K., Hammer G., Alves M., Ferreira L., Tovar E., Vasques F. (2001b). Mobility Management in RFieldbus. White Paper of the RFieldbus Project. Version 1.0.

Koulamas C., Lekkas A., Papadopoulos G., Kalivas G. Koubias S. (2001a). Delay Performance of Radio Physical Layer Technologies as Candidates for Wireless. Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Luan les Pins, France, pp. 133-142.

Kunert O. (1997). Interconnecting Field-buses through ATM. Proc. of the 2nd IEEE Int. Workshop on Factory Communication Systems – WFCS’97, Barcelona, Spain, pp. 223-229.

Kweon, S., Shin, K. (2000). Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing. Proceedings of the Real-time Technology and Application Symposium.

Lawton M. (2001). A new wave in industrial communication. http://evolution.skf.com

Leblanc C. (2000). The future of industrial networking and connectivity. Dedicated Systems Magazine, http://www.dedicated-systems.com.

Lederhofer A., Tovar E., Alves M. (1996). Using MMS is not Always Difficult: The Case of a Shop-floor Monitoring Application. IEEE-SMC Symposium on Discrete Events and Manufacturing Engineering - CESA'96 (IMACS Multiconference), July 9-12, Lille, France, pp. 490-497.

Lee K., Lee S. (2001). Integrated network of PROFIBUS-DP and IEEE 802.11 Wireless LAN with Hard Real-Time Requirement. In Proc. of the IEEE International Symposium on Industrial Electronics (ISIE'01), Pusan, Korea, pp. 1484-1489.

Lee S., Lee K., Harashima F., Lee M. (2002). Integration of Mobile Vehicles for Automated Material Handling using Profibus and IEEE 802.11 Networks. IEEE Transactions on Industrial Electronics, Volume 49, Number 3.

Li M. (1996). Real-Time Communication in an Industrial Network – PROFIBUS. PhD Thesis nº1586. École Polytechnique Fédérale de Lausanne (EPFL).

Li M., Stoeckli L. (1996). The Time Characteristics of Cyclic Service in PROFIBUS. In Proc. of the 10th International Parallel Processing Symposium, Honolulu, Hawaii, pp. 1781-1786.

Lo Bello L., Mirabella O. (1999). A Fault Tolerance Analysis of PROFIBUS Systems by Means of Generalised Stochastic Petri Nets. In Proc. of the 25th Annual Conference of the IEEE Industrial electronics Society (IECON'99), San Jose, USA, pp. 1210-1215.

Lo Bello L., Mirabella O. (2001b). A Multi-Ring Scheduling Strategy for PROFIBUS Networks. In Proc. of the 27th Annual Conference of the IEEE Industrial electronics Society (IECON'01), Denver, USA, pp. 2144-2148.

Lo Bello L., Mirabella O. (2001a). Design issues for Ethernet in Automation. Proceedings of Emerging Technologies and Factory Automation (ETFA'01), pp. 213, 221.

Marcos M., López A., Orive D. (2000). On the Analysis and Simulation of Temporal Behaviour of Real-Time Distributed Systems Using PROFIBUS. In Proc. of the 3rd IEEE International Workshop on Factory Communication Systems (WFCS'00), Porto, Portugal, pp. 179-186.

Marcos M., Orive D., Artaza F. (1997). On the Design and Development of a Gateway between MAP/MMS and PROFIBUS/FMS. Proc. of the 2nd IEEE International Workshop on Factory Communication Systems, Barcelona, Spain, pp. 349-353.

Miaoudakis A., Koukourgiannis A., Lekkas A., Kalivas G., Papadopoulos G., Kutschenreuter M., Coston P., Beziel L., Pacheco F., Papadopoulos G., Pinho L., Pipinis H., Rebakos N., Speckmeier P., Tovar E., Vasques F. (2000). Assessment and Selection of the Radio Technology. Deliverable D1.2. RFieldbus project IST-1999-11316.

Mock M., Schemmer S., Nett E. (2000). Evaluating a Wireless Real-Time Communication Protocol on Windows NT and WaveLAN. Proc of the 3rd IEEE International Workshop on Factory Communication Systems, Porto, Portugal, pp. 247-254.

Monforte S., Alves M., Vasques F., Tovar E. (2000a). Designing Real-Time Systems Based on Mono-Master PROFIBUS-DP Networks. In Proc. of the 16th IFAC Workshop on distributed Computer Control Systems, Sydney, Australia, pp. 36-43.

Monforte S., Alves M., Vasques F., Tovar E. (2000b). Segmentation Aspects in Profibus Technical Report HURRAY-TR-0019, May 2000.

Montuschi, P., Ciminiera. L. and Valenzano, A. (1992). Time Characteristics of IEE802.4 Token Bus Protocol. In IEE Proceedings, Vol. 139, No. 1, pp. 81-87.

Morel, P., Muralt, R., and Decotignie, J.-D. (1995). A Wireless Extension for Fieldbus. Proceedings of the 2nd International Conference on Industrial Automation, Nancy, France, pp. 275-279.

Neumann P., Iwanitz F. (1997). Integration of Fieldbus Systems into Distributed Object-Oriented Systems. Proc. of the 2nd IEEE International Workshop on Factory Communication Systems, Barcelona, Spain, pp. 247-253.

Pacheco F., Tovar E., Kalogeras A., Pereira N. (2001). Supporting Internet Protocols in Master-Slave Fieldbus Networks. Proceedings of the 4th IFAC International Conference on Fieldbus Systems and Their Applications (FET'2001), Nancy, France, pp. 260-266.

Palenski P., Sauter T. (2000). Security Considerations for FAN-Internet Connections. Proc. of the 3rd IEEE International Workshop on Factory Communication Systems, Porto, Portugal, pp. 27-35.

Pinto J. (1999). Fieldbus: A Neutral Instrumentation Vendor’s Perspective. Available at http://www.jimpinto.com. First published: ISA Proceedings 1994 and Intech July 1995. Updated December 1999.

Pradhan P., Chiueh T.-C. (1998). Real-Time Guarantees over Wired/Wireless LANs. Proc. of the 4th IEEE Real-Time Technology and Applications Symposium (RTAS’98), Denver, USA, pp. 29-38.

Pratl G., Lobachov M., Sauter T. (2001). Highly Modular Gateway Architecture for Fieldbus/Internet Connections. Proc of the 4th IFAC International Conference on Fieldbus Systems and their Applications (FeT’2001), Nancy, France, pp.267-273.

Renner K. M. (1999). Industrial Java Technologies... dot.com the Industrial Enterprise. Proc. of the Java Technology for Networked, Real-Time and Embedded Technologies (JISA 1999) Conference, 1999.

Rockwell Automation (2000). Making Sense of e-Manufacturing: a Roadmap for Manufacturers. White Paper. http://www.automation.rockwell.com/whitepaper/whitepaper.html.

Rüping S., Vonnhame E., Jasperneite J. (1999). Analysis of Switched Ethernet Networks with Different Topologies Used in Automation Systems. In Proceedings of Fieldbus Conference (FeT’99), Magdeburg, Germany, pp. 351-358, Springer-Verlag.

Schiffer, V. (2001). The CIP family of fieldbus protocols and its newest member - Ethernet/IP. Proceedings of Emerging Technologies and Factory Automation, pp. 377-384.

Schmid U. (1999). Basic Features of the Wireline/Wireless Factory/Facility Fieldbus. Technical Report 183/1-95, Dep. of Automation, Technical University of Vienna.

Sempere P., Victor M., Blanes J. (2001). Image Transport System in PROFIBUS Networks. In Proc. of the 4th IFAC International Conference on Fieldbus Systems and their Applications (FeT’2001), Nancy, France, pp. 24-31.

Siemens AG (2000). SIMATIC DP/PA Bus Coupler Manual. Edition 4.

Soucek S., Russ G., Tamarit C. (2000). The Smart Kitchen Project – An Application of Fieldbus Technology to Domotics. Technical Report, Technical University of Vienna, Austria.

Sveda M., Zezulka F. (1997). Interconnecting Low-Level Fieldbuses. Proc of the 23rd Euromicro Conference, Budapest, Hungary, pp. 614-620.

Tovar E., Cardoso A. (1995). On Mapping a Fieldbus Network into a MMS Server. Proc. of the 2nd International Conference on Industrial Automation, Nancy, France, pp. 267-274.

Tovar e., Vasques F. (1999a). Real-Time Fieldbus Communications Using PROFIBUS Networks. In IEEE Transactions on Industrial Electronics, Vol. 46, nº6, pp. 1241-1251.

Tovar e., Vasques F. (1999b). Cycle Time Properties of the PROFIBUS Timed Token Protocol. In Computer Communications, Elsevier Science, nº22, pp. 1206-1216.

Tovar E., Vasques F., Pacheco F., Ferreira L. (2001). Industrial Multimedia over Factory-Floor Networks. Proceedings of the 10th IFAC Symposium on Information Control Problems in Manufacturing (INCOM ’01), Vienna, Austria.

Tovar, E., Vasques, F. (2000). Non Pre-Emptive Scheduling of Messages on SMTV Token-Passing Networks. In Proc. of the 12th IEEE Euromicro Conference on Real-Time Systems (ECRTS’00), Stockholm, Sweden, pp. 209-218.

Tovar, E., Vasques, F. and Burns, A. (1999). Supporting Real-Time Distributed Computer-Controlled Systems with Multi-hop P-NET Networks. Control Engineering Practice, Vol. 7, No. 8, pp. 1015-1025, Pergamon, Elsevier Science, August 1999.

Varadarajan, S. (2001) Experiences with the EtheReal: a fault tolerant real-time Ethernet switch. Proceedings of Emerging Technologies and Factory Automation, pp. 183-194.

Varghese G., Perlman R. (1990). Transparent Interconnection of Incompatible Local Area Networks Using Bridges. In IEEE Journal on Selected Areas in Communications, nº8 (1), pp. 42-48.

Vasques F. (1996). Sur l'integration de mécanismes d'ordonnancement et de communication dans la sous-couche MAC de reseaux locaux de temp-réel. PhD thesis, Toulouse University, France.

Vasques F., Juanole G. (1994). Pre-run-time Schedulability Analysis in Fieldbus Networks. In Proc. of the 20th Annual Conference of the IEEE Industrial electronics Society (IECON'94), Bologna, Italy, pp. 1200-1204.

Wiberg P.-A., Bilstrup U. (2001). Wireless Technology in Industry – Applications and User Scenarios. Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Luan les Pins, France, pp. 123-131.

Wiberg P.-A., Svensson B. (2000). Real-Time Mobile Communication. Technical Report, Centre for Computer Systems Architecture, Halmstad University, Sweden.

Willig A. (1997). A MAC Protocol and a Scheduling Approach as Elements of a Lower Layers Architecture in wireless Industrial LANs. Proc. of the 2nd IEEE International Workshop on Factory Communication Systems, Barcelona, Spain, pp. 139-148.

Willig A. (1999). Analysis of the PROFIBUS Token Passing Protocol over Error Prone Links. In Proc. of the 25th Annual Conference of the IEEE Industrial electronics Society (IECON'99), San Jose, USA, pp. 1246-1252.

Willig A., Wolisz A. (2001). Ring Stability of the PROFIBUS Token Passing Protocol over Error Prone Links. In IEEE Transactions on Industrial Electronics, vol. 48, nº5, pp. 1025-1033.

Wollschlaeger M., (1997). Intranet based Management of Fieldbus Systems. Technical Report, RFR-Principal Consultants Inc.

Wunderlich H., Schwab M., Fredriksson L.-B. (2000). Opening Bluetooth for Technical Tasks – Possibilities and Challenges for Automotive Applications. Proc. of the Bluetooth Conference, Monte Carlo.

Zheng, Q. and Shin, K. (1995). Synchronous Bandwidth Allocation in FDDI Networks. In IEEE Transactions on Parallel and Distributed Systems, Vol. 6, No. 12, pp. 1332-1338.

Zuberi, K, and Shin, K. (1997). Scheduling Messages on Controller Area Network for Real-Time CIM Applications. In IEEE Transactions on Robotics and Automation, Vol. 13, No. 2, pp. 310-314.

Annex A

Object-Oriented Model of the Communication Network

This annex describes an object-oriented model of the hybrid wired/wireless network architecture discussed in Chapters 4 and 5. This graphical model depicts the different components/objects of the Communication Network and the relationship between them.

A.1. Basics on the Object-Oriented Analysis Method

In order to clarify the relationship between all the elements that compose the Communication Network, the object-oriented analysis method (OOA) described in (Coad and Yourdon, 1991) is used. This OOA model is structured in five layers, which gradually present more detail (Figure A.1).

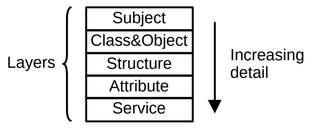


Figure A.1: Layers of the OOA modelling technique

Additionally, there are several symbols to represent the different entities of the OOA graphical model (Figure A.2).

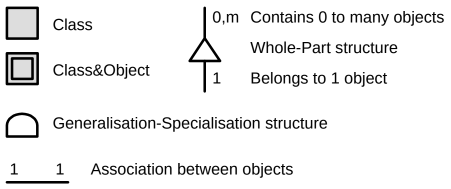

Class
Class&Object
0,m Contains 0 to many objects
Whole-Part structure
1 Belongs to 1 object
Generalisation-Specialisation structure
1 1 Association between objects

Figure A.2: Symbols of the OOA modelling technique

A double border box represents a Class and its Objects (instances). A variation of the “Class&Object” symbol is the “Class” symbol, represented by a single border box. This symbol is used to represent a generalisation Class whose corresponding Objects are portrayed by their specialisations which have Class&Object symbols (Figure A.3a). An instance connection (Figure A.2) represents an association between Objects and is represented by a line drawn between (individual) Objects. For instance in Figure A.3b, one car is associated to one owner, while one owner may have more than one car.

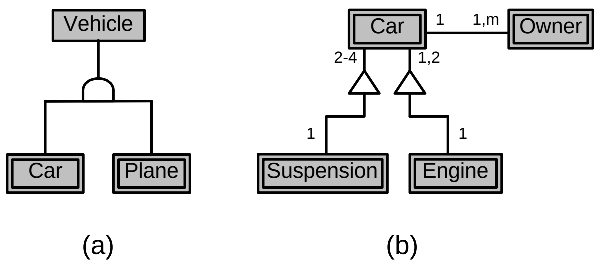


Figure A.3: Generalisation-specialisation and whole-part structures

The OOA modelling technique defines two types of structures: generalisation-specialisation and whole-part. The former type is used to distinguish between Classes (Figure A.3a). The latter tries to reflect that a whole has a number of parts. For instance, the whole-part structure depicted in Figure A.3b shows that a car can have one or two engines (hybrid) and two to four independent suspension systems. Obviously, one suspension system or one engine belongs to one (and only one) car.

A.2. Object-Oriented Model of the Communication Network

Figure A.4 presents a graphical model for the Communication Network with two (Class&Object and Structure) of the five layers defined in the OOA modelling technique. Additionally, the associations between objects are also presented (belong to the Attribute Layer). The abbreviations (Class names) are the ones proposed in Chapter 5.

A Communication Network (*N*) may be composed by one or more Communication Domains (*D*). A Communication Domain is always associated to one Physical Medium (*M*), while one Physical Medium may be associated to several (0, *m*) Communication Domains. A Communication Domain is specialised in a Wired Domain (WRD) and in a Wireless Domain (WLD). WRDs can be further specialised in MWRDs while WLDs may be of Ad-hoc (AWLD) or Structured (SWLD) types. Each AWLD is associated to one Ad-hoc Radio Cell (ARC) and vice-versa. Similarly, each SWLD is associated to one Structured Radio Cell (SRC).

Structuring Intermediate Systems (SIS) are part of a single Communication Domain (SWLD), while LIS and SLIS always belong to two Communication Domains. Wireless End Systems (WLES or MWLES) and Wired End Systems (WRES) are always associated to a number (1, *m*) of Message Streams (S). Each Message Stream has an ES acting as initiator and another ES acting as the responder, thus associated with two (different) ESs.

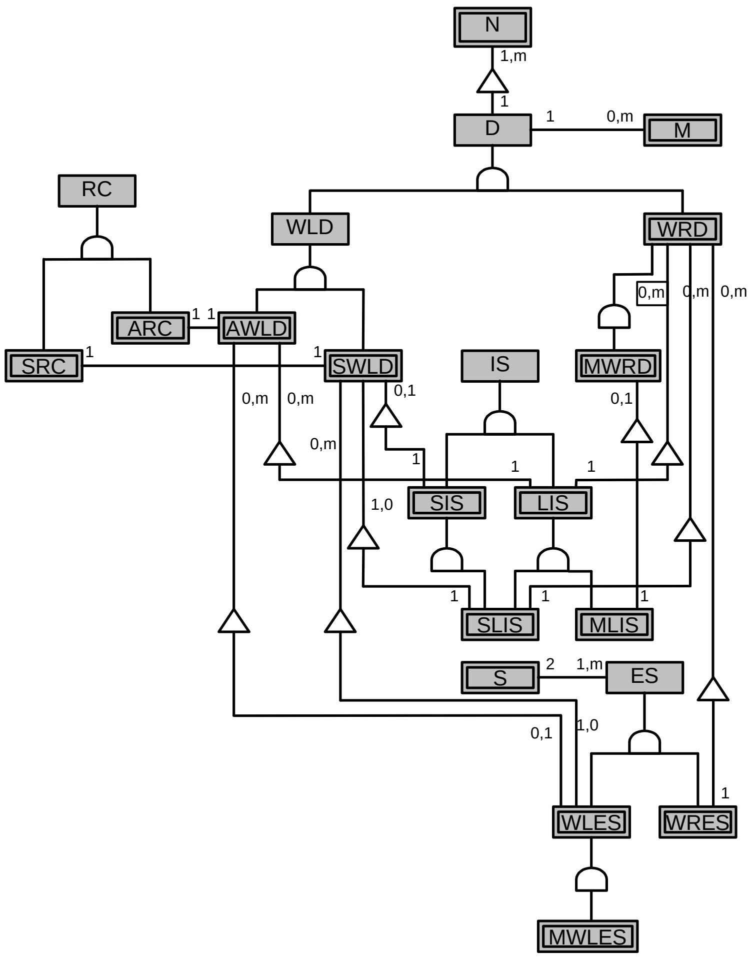


Figure A.4: Object-oriented model of the Communication Network

Annex B

Simplified Algorithm for the Computation of T_{ID1} and T_{ID2}

This annex presents a simplified algorithm intended to evaluate the idle time parameters, assuming some simplifications to the model presented in Chapter 6.

B.1 Simplifications Assumed

In order to avoid the need for the Intermediate Systems to decode the type of DLL PDU, both "minimum" idle times should be set to the same value ($T_{ID1m} = T_{ID2m} = T_{IDm}$). Thus, T_{IDm} is the idle time that will be used by all Intermediate Systems, when relaying traffic (PDUs) from one port to the other.

The methodology presented in Sections 6.3-7 permits to set both idle time parameters in a per-station basis, taking into account all possible transactions (message streams) for that master station. In this sense, each master station in the network would have a unique pair (T_{ID1}, T_{ID2}) of idle time parameter values.

For the sake of simplicity, a simplified algorithm that returns the same idle time parameter values for all masters ESs in a given Physical Medium (therefore, in a per-Physical Medium basis) is presented. Therefore, instead of considering the particular set of message streams for each master station, the algorithm assumes a worst-case scenario where maximum and minimum PDU lengths for the (overall) Communication Network are considered. This requires the definition of the following additional Communication Network parameters (Table B.1).

Table B.1: Communication Network-specific parameters

Description	Symbol
Maximum length of DLL request PDU	$L_{req}^{\max}$
Maximum length of DLL response PDU	$L_{resp}^{\max}$
Minimum length of DLL request PDU	$L_{req}^{\min}$
Minimum length of DLL response PDU	$L_{resp}^{\min}$

Moreover, acknowledged and unacknowledged DLL request PDUs are considered to have the same maximum and minimum lengths.

An algorithm (presented in pseudo-code) that evaluates the values for T_{ID1} and T_{ID2} is proposed next.

B.2 Simplified Algorithm for the Computation of T_{ID1} and T_{ID2}

```
Begin
/* Read Communication Network-specific parameters */
input($L_{token}$ ,$L^{max}_{req}$ , $L^{max}_{resp}$ , $L^{min}_{req}$ , $L^{min}_{resp}$ ,$t^{min}_{rt}$ , $T_{IDm}$ , d, $t_{rd}$ ,nm)
/* Set minimum idle time parameters to the same value */
$T_{ID1m}$= $T_{ID2m}$= $T_{IDm}$
For i=1 to nm /* For every physical media */
    /* Read Physical Medium-specific parameters *)
    input($r^i$, $l^i_H$, $l^i_T$, $k^i$, $o^i$)
    For j=1 to nm        /* For every Physical Medium */
        if j <> i then   /* other than i (j¹i) */
            /* Set DLL PDU lengths to maximise the idle time */
            If $\frac{d+k^j}{r^j} > \frac{d+k^i}{r^i}$ then
                $L_{req(l-1)} = L_{req(l)} = L^{\max}_{req}, L_{resp(l-1)} = L^{\max}_{resp}$
            else
                $L_{req(l-1)} = L^{\min}_{req}, L_{req(l)} = L_{token}, L_{resp(l-1)} = L^{\min}_{resp}$
            endif
            /* Compute $t^{i®j}_{ID1G+}$ (Eq. 6.6) and $t^{i®j}_{ID1D+}$ (Eq. 6.21) */
            $t^{i\to j}_{ID1\Gamma+} \geq C^j_{req(l-1)} - C^i_{req(l-1)} + C^j_{resp(l-1)} - C^i_{resp(l-1)} + 2 \cdot t^j_{ID1m} - t^i_{ID1m} - t^{\min}_{rt} +$
                $+ t^{i\to j}_{srreq(l-1)} - t^{i\to j}_{srreq(l)} +$
                $+ \max\{t^{i\to j}_{srresp(l-1)} - t^{i\to j}_{srreq(l-1)} + C^i_{req(l-1)} - C^j_{req(l-1)} + t^{\min}_{rt} - t^j_{ID1m}, 0\}$
            $t^{i\to j}_{ID1\Delta+} \geq t^{i\to j}_{srtoken} - t^{i\to j}_{srreq(l)} + C^j_{token} - C^i_{token} + t^j_{ID1m} - t^i_{ID1m}$
            $t^{i\to j}_{ID1+} = \max\{t^{i\to j}_{ID1\Gamma+}, t^{i\to j}_{ID1\Delta+}\}$
            /* Compute $t^{i®j}_{ID2+}$ using Eq. 6.31 */
            $t^{i\to j}_{ID2+} \geq t^{i\to j}_{srreq(l-1)} - t^{i\to j}_{srreq(l)} + C^j_{req(l-1)} - C^i_{req(l-1)} + t^j_{ID2m} - t^i_{ID2m}$
        endif /* if j <> i */
    endfor /* For j */
    $t^i_{ID1+} = \max\{t^{i\to j}_{ID1+}\}$
    $t^i_{ID2+} = \max\{t^{i\to j}_{ID2+}\}$
    $T^i_{ID1} = T_{ID1m} + \lceil t^i_{ID1+} \cdot r^i \rceil$
    $T^i_{ID2} = T_{ID2m} + \lceil t^i_{ID2+} \cdot r^i \rceil$
endfor /* For i */
end.
```

Annex C

On the Conditions for no Queuing Delays in the Intermediate Systems

The inserted idle time guarantees that there are no increasing queuing delays (no congestion) in the Intermediate Systems. Request PDUs may be affected by queuing delays from the second IS on, between initiator and responder. In order to compute the worst-case system turnaround time for a message transaction (t_{st}), it is necessary to compute the worst-case total queuing delay (Q) affecting a request PDU, from the initiator up to the responder. The results presented in this annex are of paramount importance, since they prove that for a worst-case situation where a request PDU (transaction l) is preceded by an infinite sequence of equal length PDUs, Q can be computed considering just the previous transaction (l-1).

C.1 Case 1: Transaction *l*-1 is Acknowledged (Req/Resp)

In this section, it is proved that if $L_{req(l-1)} = L_{resp(l-1)} = L_{req(l)}$, then the request PDU of transaction l has no queuing delay (in none of the IS up to the responder). This occurs if the request PDU of transaction (l-1) has no queuing delay and $G^i_a \geq G^i_b$, for any $i \in I$.

By induction, if $G^1_a \geq G^1_b$ is true, if it is proved that $G^{n+1}_a \geq G^{n+1}_b$ considering true that $G^n_a \geq G^n_b$, then it is true that $G^i_a \geq G^i_b$, for any $i \in I$.

To prove that $G^1_a \geq G^1_b$ is to prove that (see Eq. (7.12)):

$$C^1_{req(l-1)} + t_{rt} + C^1_{resp(l-1)} + t^1_{ID1m} + t^1_{ID1+} + t^{1\to 2}_{srreq(l)} + t_{rd}$$

$$\geq$$

$$\max\left\{C^1_{req(l-1)} + t_{rt} + t^{1\to 2}_{srresp(l-1)} + t_{rd},\ t^{1\to 2}_{srreq(l-1)} + t_{rd} + C^2_{req(l-1)} + t^2_{ID1m}\right\} + C^2_{resp(l-1)} + t^2_{ID1m}$$

which results in:

$$\begin{aligned} t^1_{ID1+} \geq\ & C^2_{req(l-1)} - C^1_{req(l-1)} + C^2_{resp(l-1)} - C^1_{resp(l-1)} + 2 \cdot t^2_{ID1m} - t^1_{ID1m} - t_{rt} + \\ & + t^{1\to 2}_{srreq(l-1)} - t^{1\to 2}_{srreq(l)} + \\ & + \max\left\{t^{1\to 2}_{srresp(l-1)} - t^{1\to 2}_{srreq(l-1)} + C^1_{req(l-1)} - C^2_{req(l-1)} + t_{rt} - t^2_{ID1m},\ 0\right\} \end{aligned} \qquad (C.1)$$

Since, from Eq. (6.7) that $t^{1}_{ID1+}\ ^{3}\ t^{1®2}_{ID1+}$ and taking into account that, from Eq. (6.6):

$$\begin{aligned} t^{1\to 2}_{ID1+} \geq{} & C^{2}_{req(l-1)} - C^{1}_{req(l-1)} + C^{2}_{resp(l-1)} - C^{1}_{resp(l-1)} + 2\cdot t^{2}_{ID1m} - t^{1}_{ID1m} - t_{rt} + \\ & + t^{1\to 2}_{srreq(l-1)} - t^{1\to 2}_{srreq(l)} + \\ & + \max\left\{ t^{1\to 2}_{srresp(l-1)} - t^{1\to 2}_{srreq(l-1)} + C^{1}_{req(l-1)} - C^{2}_{req(l-1)} + t_{rt} - t^{2}_{ID1m},\ 0 \right\} \end{aligned}$$

Then, Eq. (C.1) is true, that is what we wanted to prove. Note that this result was expected, since the inserted idle time is defined in a way to prevent queuing delays in the first IS.

Now let us check if $G^{n+1}{}_{a} \geq G^{n+1}{}_{b}$ is true (and in which conditions), considering true that:

$$\Gamma^{n}_{a} \geq \Gamma^{n}_{b} \tag{C.2}$$

By definition (Eq. (7.14)),

$$\Gamma^{n+1}_{a} = \max\left\{\Gamma^{n}_{a}, \Gamma^{n}_{b}\right\} + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \tag{C.3}$$

Taking into account Eq. (C.2), it is true that

$$\Gamma^{n+1}_{a} = \Gamma^{n}_{a} + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \tag{C.4}$$

Considering that it is also true that:

$$\Gamma^{i}_{a} \geq \Gamma^{i}_{b},\quad \forall \mathrm{i} \in \{1,\ldots,\mathrm{n}-1\} \tag{C.5}$$

then,

$$\Gamma^{n+1}_{a} = \Gamma^{1}_{a} + \sum_{i=2}^{n+1}\left(t^{i\to i+1}_{srreq(l)} + t_{rd}\right) \tag{C.6}$$

By definition (Eq. (7.14)),

$$\Gamma^{n+1}_{b} = \max\left\{\begin{array}{l} \Gamma^{n}_{b} - C^{n+1}_{resp(l-1)} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{srresp(l-1)} + t_{rd}, \\ \sum_{j=1}^{n+1}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{n+2}_{req(l-1)} + t^{n+2}_{ID1m} \end{array}\right\} + C^{n+2}_{resp(l-1)} + t^{n+2}_{ID1m} \tag{C.7}$$

Let us first analyse the case where:

$$\Gamma^{n+1}_{b} = \sum_{j=1}^{n+1}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{n+2}_{req(l-1)} + t^{n+2}_{ID1m} + C^{n+2}_{resp(l-1)} + t^{n+2}_{ID1m} \tag{C.8}$$

The intention is to prove that:

$$\Gamma^{1}_{a} + \sum_{j=2}^{n+1}\left(t^{j\to j+1}_{srreq(l)} + t_{rd}\right) \geq \sum_{j=1}^{n+1}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{n+2}_{req(l-1)} + t^{n+2}_{ID1m} + C^{n+2}_{resp(l-1)} + t^{n+2}_{ID1m} \tag{C.9}$$

If the following simplification is assumed:

$$L_{resp(l-1)} = L_{req(l-1)} = L_{req(l)} = L \Rightarrow \qquad \text{(C.10)}$$
$$\Rightarrow \begin{cases} C^{j}_{resp(l-1)} = C^{j}_{req(l-1)} = C^{j}_{req(l)} = C^{j}, \quad \forall \mathrm{j} \in \{1,...,\mathrm{ndp}\} \\ t^{i\to i+1}_{srresp(l-1)} = t^{i\to i+1}_{srreq(l-1)} = t^{i\to i+1}_{srreq(l)} = t^{i\to i+1}_{sr}, \quad \forall \mathrm{i} \in \mathrm{I} \end{cases}$$

then Eq. (C.9) turns into:

$$\Gamma^{1}_{a} + \sum_{j=2}^{n+1}\left(t^{j\to j+1}_{sr} + t_{rd}\right) \geq \sum_{j=1}^{n+1}\left(t^{j\to j+1}_{sr} + t_{rd}\right) + C^{n+2} + t^{n+2}_{ID1m} + C^{n+2} + t^{n+2}_{ID1m} \Leftrightarrow \qquad \text{(C.11)}$$
$$\Leftrightarrow \Gamma^{1}_{a} \geq t^{1\to 2}_{sr} + t_{rd} + 2\cdot C^{n+2} + 2\cdot t^{n+2}_{ID1m}$$

Taking into account Eqs. (7.12) and (C.10):

$$\Gamma^{1}_{a} = C^{1} + t_{rt} + C^{1} + t^{1}_{ID1m} + t^{1}_{ID1+} + t^{1\to 2}_{sr} + t_{rd} = \qquad \text{(C.12)}$$
$$= 2\cdot C^{1} + t_{rt} + t^{1}_{ID1m} + t^{1}_{ID1+} + t^{1\to 2}_{sr} + t_{rd}$$

Substituting Eq. (C.12) in (C.11) stands:

$$2\cdot C^{1} + t_{rt} + t^{1}_{ID1m} + t^{1}_{ID1+} + t^{1\to 2}_{sr} + t_{rd} \geq t^{1\to 2}_{sr} + t_{rd} + 2\cdot C^{n+2} + 2\cdot t^{n+2}_{ID1m} \Leftrightarrow \qquad \text{(C.13)}$$
$$\Leftrightarrow t^{1}_{ID1+} \geq 2\cdot C^{n+2} - 2\cdot C^{1} + 2\cdot t^{n+2}_{ID1m} - t^{1}_{ID1m} - t_{rt}$$

Taking into account that, from Eqs. (6.6) and (C.10):

$$t^{1\to n+2}_{ID1+} \geq C^{n+2} - C^{1} + C^{n+2} - C^{1} + 2\cdot t^{n+2}_{ID1m} - t^{1}_{ID1m} - t_{rt} +$$
$$+ t^{1\to 2}_{sr} - t^{1\to 2}_{sr} +$$
$$+ \max\left\{t^{1\to n+2}_{sr} - t^{1\to n+2}_{sr} + C^{1} - C^{n+2} + t_{rt} - t^{n+2}_{ID1m}, 0\right\} \Rightarrow$$
$$t^{1\to n+2}_{ID1+} \geq 2\cdot C^{n+2} - 2\cdot C^{1} + 2\cdot t^{n+2}_{ID1m} - t^{1}_{ID1m} - t_{rt}$$

then. since from Eq. (6.7), t^{1}_{ID1+} ³ $t^{1®n+2}_{ID1+}$, then it is true that:

$$t^{1}_{ID1+} \geq 2\cdot C^{n+2} - 2\cdot C^{1} + 2\cdot t^{n+2}_{ID1m} - t^{1}_{ID1m} - t_{rt} \qquad \text{(C.14)}$$

That is equivalent to Eq. (C.13), what we wanted to prove.
Let us now analyse the case in Eq. (C.7) where:

$$\Gamma^{n+1}_{b} = \Gamma^{n}_{b} - C^{n+1}_{resp(l-1)} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{srresp(l-1)} + t_{rd} + C^{n+2}_{resp(l-1)} + t^{n+2}_{ID1m} \qquad \text{(C.15)}$$

The intention is to prove that:

$$\Gamma^{n}_{a} + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \geq \Gamma^{n}_{b} - C^{n+1}_{resp(l-1)} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{srresp(l-1)} + t_{rd} + C^{n+2}_{resp(l-1)} + t^{n+2}_{ID1m} \qquad \text{(C.16)}$$

Assuming the simplification in Eq. (C.10), Eq. (C.16) results in:

$$\begin{aligned}&\Gamma_a^n + t_{sr}^{n+1\to n+2} + t_{rd} \geq \Gamma_b^n - C^{n+1} - t_{ID1m}^{n+1} + t_{sr}^{n+1\to n+2} + t_{rd} + C^{n+2} + t_{ID1m}^{n+2} \Leftrightarrow \\ &\Leftrightarrow \Gamma_a^n \geq \Gamma_b^n + C^{n+2} - C^{n+1} + t_{ID1m}^{n+2} - t_{ID1m}^{n+1}\end{aligned} \tag{C.17}$$

Proving Eq. (C.17) by induction:

$$\Gamma_a^1 \geq \Gamma_b^1 + C^3 - C^2 + t_{ID1m}^3 - t_{ID1m}^2 \tag{C.18}$$

Substituting $G^1{}_a$ and $G^1{}_b$ using Eq. (7.12) stands:

$$\begin{aligned}&2\cdot C^1 + t_{rt} + t_{ID1m}^1 + t_{ID1+}^1 + t_{sr}^{1\to 2} + t_{rd} \geq \\ &\max\left\{C^1 + t_{rt} + t_{sr}^{1\to 2} + t_{rd},\ t_{sr}^{1\to 2} + t_{rd} + C^2 + t_{ID1m}^2\right\} \\ &+ C^2 + t_{ID1m}^2 + C^3 - C^2 + t_{ID1m}^3 - t_{ID1m}^2 \\ &\Leftrightarrow \\ &2\cdot C^1 + t_{rt} + t_{ID1m}^1 + t_{ID1+}^1 + t_{sr}^{1\to 2} \geq \max\left\{C^1 + t_{rt} + t_{sr}^{1\to 2}, t_{sr}^{1\to 2} + C^2 + t_{ID1m}^2\right\} + C^3 + t_{ID1m}^3\end{aligned} \tag{C.19}$$

Let us analyse the case where:

$$\begin{aligned}&2\cdot C^1 + t_{rt} + t_{ID1m}^1 + t_{ID1+}^1 + t_{sr}^{1\to 2} \geq C^1 + t_{rt} + t_{sr}^{1\to 2} + C^3 + t_{ID1m}^3 \Leftrightarrow \\ &\Leftrightarrow t_{ID1+}^1 \geq C^3 - C^1 + t_{ID1m}^3 - t_{ID1m}^1\end{aligned} \tag{C.20}$$

Since, from Eq. (6.7), $t^1{}_{ID1+}$ ³ $t^{1®3}{}_{ID1+}$ and taking into account that, from Eq. (6.6):

$$\begin{aligned}t_{ID1+}^{1\to 3} &\geq C^3 - C^1 + C^3 - C^1 + 2\cdot t_{ID1m}^3 - t_{ID1m}^1 - t_{rt} + \\ &+ t_{sr}^{1\to 3} - t_{sr}^{1\to 3} + \\ &+ \max\left\{t_{sr}^{1\to 3} - t_{sr}^{1\to 3} + C^1 - C^3 + t_{rt} - t_{ID1m}^3, 0\right\} \\ &\geq C^3 - C^1 + t_{ID1m}^3 - t_{ID1m}^1\end{aligned}$$

then it is true that:

$$t_{ID1+}^1 \geq C^3 - C^1 + t_{ID1m}^3 - t_{ID1m}^1 \tag{C.21}$$

That is what we wanted to prove (Eq. (C.20)).
Now it is necessary to check the other case, i.e. if it is true that:

$$\begin{aligned}&2\cdot C^1 + t_{rt} + t_{ID1m}^1 + t_{ID1+}^1 + t_{sr}^{1\to 2} \geq t_{sr}^{1\to 2} + C^2 + t_{ID1m}^2 + C^3 + t_{ID1m}^3 \Leftrightarrow \\ &\Leftrightarrow t_{ID1+}^1 \geq C^2 - C^1 + C^3 - C^1 + t_{ID1m}^2 + t_{ID1m}^3 - t_{ID1m}^1 - t_{rt}\end{aligned} \tag{C.22}$$

From Eq. (6.7), it is true that:

$$\begin{cases} t^{1}_{ID1+} \geq t^{1\to 2}_{ID1+} \\ t^{1}_{ID1+} \geq t^{1\to 3}_{ID1+} \end{cases} \Rightarrow 2\cdot t^{1}_{ID1+} \geq t^{1\to 2}_{ID1+} + t^{1\to 3}_{ID1+} \Leftrightarrow t^{1}_{ID1+} \geq \frac{t^{1\to 2}_{ID1+} + t^{1\to 3}_{ID1+}}{2} \tag{C.23}$$

Computing t^{1}_{ID1+} + $t^{1®3}_{ID1+}$ results in:

$$\begin{aligned} t^{1\to 2}_{ID1+} + t^{1\to 3}_{ID1+} \geq\; & 2\cdot C^{2} - 2\cdot C^{1} + 2\cdot t^{2}_{ID1m} - t^{1}_{ID1m} - t_{rt} + \\ & + 2\cdot C^{3} - 2\cdot C^{1} + 2\cdot t^{3}_{ID1m} - t^{1}_{ID1m} - t_{rt} + \\ & + \max\left\{t^{1\to 2}_{sr} - t^{1\to 2}_{sr} + C^{1} - C^{2} + t_{rt} - t^{2}_{ID1m}, 0\right\} + \\ & + \max\left\{t^{1\to 3}_{sr} - t^{1\to 3}_{sr} + C^{1} - C^{3} + t_{rt} - t^{3}_{ID1m}, 0\right\} \\ \geq\; & 2\cdot\left(C^{2} - C^{1} + C^{3} - C^{1} + t^{2}_{ID1m} + t^{3}_{ID1m} - t^{1}_{ID1m} - t_{rt}\right) + 0 + 0 \end{aligned} \tag{C.24}$$

Eqs. (C.23) and (C.24) prove that (C.22) is true.
Now, Eq. (C.17) must be proved for the general case. Assuming that is true that:

$$\Gamma^{n}_{a} \geq \Gamma^{n}_{b} + C^{n+2} - C^{n+1} + t^{n+2}_{ID1m} - t^{n+1}_{ID1m} \tag{C.25}$$

The intention is to prove that:

$$\Gamma^{n+1}_{a} \geq \Gamma^{n+1}_{b} + C^{n+3} - C^{n+2} + t^{n+3}_{ID1m} - t^{n+2}_{ID1m} \tag{C.26}$$

Considering Eq. (C.7) and the simplification expressed in (C.10),

$$\Gamma^{n+1}_{b} = \max\left\{ \begin{array}{l} \Gamma^{n}_{b} - C^{n+1} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{sr} + t_{rd}, \\ \sum\limits_{i=1}^{n+1}\left(t^{i\to i+1}_{sr} + t_{rd}\right) + C^{n+2} + t^{n+2}_{ID1m} \end{array} \right\} + C^{n+2} + t^{n+2}_{ID1m} \tag{C.27}$$

Starting by addressing the case where:

$$\Gamma^{n+1}_{b} = \Gamma^{n}_{b} - C^{n+1} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{sr} + t_{rd} + C^{n+2} + t^{n+2}_{ID1m} \tag{C.28}$$

then, it must be proved that:

$$\begin{aligned} & \Gamma^{n+1}_{a} \geq \Gamma^{n}_{b} - C^{n+1} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{sr} + t_{rd} + C^{n+2} + t^{n+2}_{ID1m} \\ & + C^{n+3} - C^{n+2} + t^{n+3}_{ID1m} - t^{n+2}_{ID1m} \Leftrightarrow \\ & \Leftrightarrow \Gamma^{n+1}_{a} \geq \Gamma^{n}_{b} + C^{n+3} - C^{n+1} + t^{n+3}_{ID1m} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{sr} + t_{rd} \end{aligned} \tag{C.29}$$

Considering Eq. (C.4),

$$\begin{aligned} & \Gamma^{n}_{a} + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \geq \Gamma^{n}_{b} + C^{n+3} - C^{n+1} + t^{n+3}_{ID1m} - t^{n+1}_{ID1m} + t^{n+1\to n+2}_{sr} + t_{rd} \Leftrightarrow \\ & \Leftrightarrow \Gamma^{n}_{a} \geq \Gamma^{n}_{b} + C^{n+3} - C^{n+1} + t^{n+3}_{ID1m} - t^{n+1}_{ID1m} \end{aligned} \tag{C.30}$$

That was admitted to be true.

Now let us address the case where:

$$\begin{aligned}\Gamma_b^{n+1} &= \sum_{i=1}^{n+1}\left(t_{sr}^{i\to i+1}+t_{rd}\right)+C^{n+2}+t_{ID1m}^{n+2}+C^{n+2}+t_{ID1m}^{n+2} = \\ &= \sum_{i=1}^{n+1}\left(t_{sr}^{i\to i+1}+t_{rd}\right)+2\cdot C^{n+2}+2\cdot t_{ID1m}^{n+2}\end{aligned} \tag{C.31}$$

Substituting Eqs. (C.6) and (C.31) in Eq. (C.26), and taking into account the simplification (C.10), results in:

$$\Gamma_a^1+\sum_{i=2}^{n+1}\left(t_{sr}^{i\to i+1}+t_{rd}\right)\geq \sum_{i=1}^{n+1}\left(t_{sr}^{i\to i+1}+t_{rd}\right)+C^{n+2}+C^{n+3}+t_{ID1m}^{n+2}+t_{ID1m}^{n+3} \tag{C.32}$$

Taking into account Eq. (C.12), Eq. (C.32) stands:

$$\begin{gathered}2\cdot C^1+t_{rt}+t_{ID1m}^1+t_{ID1+}^1+t_{sr}^{1\to 2}+t_{rd}+\sum_{i=2}^{n+1}\left(t_{srreq(l)}^{i\to i+1}+t_{rd}\right) \\ \geq \\ \sum_{i=1}^{n+1}\left(t_{sr}^{i\to i+1}+t_{rd}\right)+C^{n+2}+C^{n+3}+t_{ID1m}^{n+2}+t_{ID1m}^{n+3} \Leftrightarrow \\ t_{ID1+}^1 \geq C^{n+3}-C^1+C^{n+2}-C^1+t_{ID1m}^{n+2}+t_{ID1m}^{n+3}-t_{ID1m}^1-t_{rt}\end{gathered} \tag{C.33}$$

From Eq. (6.7),

$$\begin{cases} t_{ID1+}^1 \geq t_{ID1+}^{1\to n+2} \\ t_{ID1+}^1 \geq t_{ID1+}^{1\to n+3}\end{cases} \Rightarrow 2\cdot t_{ID1+}^1 \geq t_{ID1+}^{1\to n+2}+t_{ID1+}^{1\to n+3} \Leftrightarrow t_{ID1+}^1 \geq \frac{t_{ID1+}^{1\to n+2}+t_{ID1+}^{1\to n+3}}{2} \tag{C.34}$$

Computing $t^{1\circledR n+2}{}_{ID1+}$ + $t^{1\circledR n+3}{}_{ID1+}$ results in:

$$\begin{aligned} t_{ID1+}^{1\to n+2}+t_{ID1+}^{1\to n+3} \geq\ & 2\cdot C^{n+2}-2\cdot C^1+2\cdot t_{ID1m}^{n+2}-t_{ID1m}^1-t_{rt}+ \\ & +2\cdot C^{n+3}-2\cdot C^1+2\cdot t_{ID1m}^{n+3}-t_{ID1m}^1-t_{rt}+ \\ & +\max\left\{t_{sr}^{1\to n+2}-t_{sr}^{1\to n+2}+C^1-C^{n+2}+t_{rt}-t_{ID1m}^{n+2},0\right\}+ \\ & +\max\left\{t_{sr}^{1\to n+3}-t_{sr}^{1\to n+3}+C^1-C^{n+3}+t_{rt}-t_{ID1m}^{n+3},0\right\} \\ \geq\ & 2\cdot\left(C^{n+2}-C^1+C^{n+3}-C^1+t_{ID1m}^{n+2}+t_{ID1m}^{n+3}-t_{ID1m}^1-t_{rt}\right)+0+0\end{aligned} \tag{C.35}$$

Taking into consideration Eqs. (C.34) and (C.35), it is also true that:

$$t_{ID1+}^1 \geq C^{n+2}-C^1+C^{n+3}-C^1+t_{ID1m}^{n+2}+t_{ID1m}^{n+3}-t_{ID1m}^1-t_{rt} \tag{C.36}$$

That is what we wanted to prove (5.35).

The previous analysis allows us to state that if $L_{req(l-1)} = L_{resp(l-1)} = L_{req(l)}$, then the request PDU of transaction *l* never experiences queuing delay, from the initiator to the responder.

C.2 Case 2: Transaction *l*-1 is Unacknowledged (SDN)

In this section, it is proved that if the lengths of the request PDU of transaction *l* and of the unacknowledged request PDU of transaction *l*-1 are equal ($L_{req(l-1)} = L_{req(l)}$), then the request PDU of transaction *l* has no queuing delay (in none of the IS until reaching the responder). This situation occurs if the request PDU of transaction (*l*-1) has no queuing delay and $F^i_a \geq F^i_b$, for any *i* belonging to the ordered set of Communication Domains from the initiator to the responder.

By induction, if $F^1_a \geq F^1_b$ is true, if it is proved that $F^{n+1}_a \geq F^{n+1}_b$ considering true that $F^n_a \geq F^n_b$, then $F^i_a \geq F^i_b$, for any *i*.

Prove that $F^1_a \geq F^1_b$, i.e.:

$$\begin{aligned}&C^1_{req(l-1)} + t^1_{ID2m} + t^1_{ID2+} + t^{1\to 2}_{srreq(l)} + t_{rd} \geq t^{1\to 2}_{srreq(l-1)} + t_{rd} + C^2_{req(l-1)} + t^2_{ID2m} \Leftrightarrow \\ &\Leftrightarrow t^1_{ID2+} \geq t^{1\to 2}_{srreq(l-1)} - t^{1\to 2}_{srreq(l)} + C^2_{req(l-1)} - C^1_{req(l-1)} + t^2_{ID2m} - t^1_{ID2m}\end{aligned} \tag{C.37}$$

Since, from Eq. (6.33), $t^1_{ID2+} \geq t^{1®2}_{ID2+}$ and taking into account that, from Eq. (6.32):

$$t^{1\to 2}_{ID2+} \geq t^{1\to 2}_{srreq(l-1)} - t^{1\to 2}_{srreq(l)} + C^2_{req(l-1)} - C^1_{req(l-1)} + t^2_{ID2m} - t^1_{ID2m} \tag{C.38}$$

Then Eq. (C.37) is true, that is what we wanted to prove. Note that this result was expected, since the inserted idle time is defined in a way to prevent queuing delays in the first IS.

Now, assuming that $F^n_a \geq F^n_b$, the intention is to prove that $F^{n+1}_a \geq F^{n+1}_b$.

By definition (Eq. (7.19)):

$$\Phi^{n+1}_a = \max\left\{\Phi^n_a, \Phi^n_b\right\} + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \tag{C.39}$$

Taking into account our assumption, it is true that

$$\Phi^{n+1}_a = \Phi^n_a + t^{n+1\to n+2}_{srreq(l)} + t_{rd} \tag{C.40}$$

Considering that it is also true that:

$$\Phi^i_a \geq \Phi^i_b, \quad \forall \mathrm{i} \in \{1,\ldots,\mathrm{n-1}\} \tag{C.41}$$

then,

$$\Phi^{n+1}_a = \Phi^1_a + \sum_{j=2}^{n+1}\left(t^{j\to j+1}_{srreq(l)} + t_{rd}\right) \tag{C.42}$$

By definition (Eq. (7.19)),

$$\Phi^{n+1}_b = \sum_{j=1}^{n+1}\left(t^{j\to j+1}_{srreq(l-1)} + t_{rd}\right) + C^{n+2}_{req(l-1)} + t^{n+2}_{ID2m} \tag{C.43}$$

Therefore, the intention is to prove that:

$$\Phi_a^1 + \sum_{j=2}^{n+1}\left(t_{srreq(l)}^{j\to j+1} + t_{rd}\right) \geq \sum_{j=1}^{n+1}\left(t_{srreq(l-1)}^{j\to j+1} + t_{rd}\right) + C_{req(l-1)}^{n+2} + t_{ID2m}^{n+2} \tag{C.44}$$

If the following simplification is assumed:

$$L_{req(l-1)} = L_{req(l)} = L \Rightarrow \begin{cases} C_{req(l-1)}^{i} = C_{req(l)}^{i} = C^{i}, \quad \forall \mathrm{i} \in \{1,...,\mathrm{ndp}\} \\ t_{srreq(l-1)}^{j\to j+1} = t_{srreq(l)}^{j\to j+1} = t_{sr}^{j\to j+1}, \quad \forall \mathrm{j} \in \{1,...,(\mathrm{ndp}-1)\} \end{cases} \tag{C.45}$$

Then Eq. (C.44) results in:

$$\begin{gathered} \Phi_a^1 + \sum_{j=2}^{n+1}\left(t_{sr}^{j\to j+1} + t_{rd}\right) \geq \sum_{j=1}^{n}\left(t_{sr}^{j\to j+1} + t_{rd}\right) + C^{n+2} + t_{ID2m}^{n+2} \Leftrightarrow \\ \Leftrightarrow \Phi_a^1 \geq t_{sr}^{1\to 2} + t_{rd} + C^{n+2} + t_{ID2m}^{n+2} \end{gathered} \tag{C.46}$$

Taking into account the definition in Eq. (7.17) and the simplification in (C.45):

$$\Phi_a^1 = C^1 + t_{ID2m}^1 + t_{ID2+}^1 + t_{sr}^{1\to 2} + t_{rd} \tag{C.47}$$

Substituting Eq. (C.47) in (C.46) stands:

$$\begin{gathered} C^1 + t_{ID2m}^1 + t_{ID2+}^1 + t_{sr}^{1\to 2} + t_{rd} \geq t_{sr}^{1\to 2} + t_{rd} + C^{n+2} + t_{ID2m}^{n+2} \Leftrightarrow \\ \Leftrightarrow t_{ID2+}^1 \geq C^{n+2} - C^1 + t_{ID1m}^{n+2} - t_{ID2m}^1 \end{gathered} \tag{C.48}$$

Taking into account that, from Eq. (6.32):

$$t_{ID2+}^{1\to n+2} \geq t_{srreq(l-1)}^{1\to n+2} - t_{srreq(l)}^{1\to n+2} + C_{req(l-1)}^{n+2} - C_{req(l-1)}^{1} + t_{ID2m}^{n+2} - t_{ID2m}^{1} \tag{C.49}$$

Considering the simplification in (C.45) and that $t^{1}_{ID2+} \geq t^{1\text{®}n+2}_{ID2+}$ (from Eq. 6.33), then it is true:

$$t_{ID2+}^1 \geq t_{ID2+}^{1\to n+2} \geq C^{n+2} - C^1 + t_{ID2m}^{n+2} - t_{ID2m}^1 \tag{C.50}$$

That is equivalent to (C.48), what we wanted to prove.

The previous analysis allows to conclude that if $L_{req(l-1)} = L_{req(l)}$, then the request PDU of transaction *l* never experiences queuing delay, from the initiator to the responder.

Annex D

Algorithms for the Computation of t_{st}, T_{SL1} and T_{SL2}

This annex presents two pseudo-code algorithms. The first one concerns the computation of the worst-case system turnaround time (t_{st}), the worst-case duration of message transactions, and the first component of the PROFIBUS Slot Time parameter - T_{SL1}. The second algorithm concerns the computation of the second component of Slot Time parameter - T_{SL2}.

D.1 Computation of t_{st} , C_{ack} and T_{SL1}

The following algorithm (presented in pseudo-code) for the computation of T_{st} , C_{ack} and T_{SL1} is proposed:

Algorithm D.1: Computation of t_{st} , C_{ack} and T_{SL1}

```
Begin
/* Define the Stream structure type */
S_type =  record
          ndp: integer    /* number of Comm. Domains in the path */
                          /* between initiator and responder */
          path: array[10] of integer /* physical media in the path */
          Lreq: integer   /* length of the request PDU */
          Lresp: integer  /* length of the response PDU */
          tstn: real      /* system turnaround time no queuing */
          Gama_a: array[10] of real
          Gama_b: array[10] of real
          q_Gama: array[10] of real /* queuing Gama in each domain */
          Qgama: real     /* total queuing Gama in the path */
          Fi_a: array[10] of real
          Fi_b: array[10] of real
          q_Fi: array[10] of real /* queuing Fi in each domain */
          QFi: real       /* total queuing Fi in the path */
          Q: real         /* total queuing in the path */
          tst: real       /* total system turnaround time */
          Duration: real  /* duration of the message transaction */
          endrecord
/* Define the Stream structure variable */
S: array[100] of S_type
/* Call the Idle_Time procedure */
Call Idle_Time
/* Read Stream-related parameters */
input(ns)                 /* number of streams */
For i=1 to ns             /* For every Stream */
    input(S[i].ndp)
    for j=1 to ndp
        input(S[i].path[j])
    endfor /* j */
    input(S[i].Lreq)
    input(S[i].Lresp)
endfor /* i */
/* Computation of the worst-case system turnaround time */
For i=1 to ns             /* For every Stream */
    if S[i].ndp = 1 then  /* if there are no IS in the path */
        S[i].tstn = trt_max/* System’s = Responder’s turnaround t */
        S[i].Q = 0        /* no queuing */
    else                  /* one or more IS in the path */
        /* Computation of tstn */
        $S[i].tstn = \sum_{i=1}^{ndp-1}\left(t_{srLreq}^{i\to i+1} + t_{rd}\right) + C_{Lreq}^{n} + t_{rt}^{\max} + \sum_{i=ndp}^{2}\left(t_{srLresp}^{i\to i-1} + t_{rd}\right) - C_{Lreq}^{1}$
        /* Computation of the queuing delays*/
        if S[i].ndp = 2 then  /* only one IS in the path */
           S[i].Q = 0         /* no queuing in first IS */
        else                  /* two or more IS in the path */
           /* Computation of Qgama */
           /* Computation of Gama for the first IS */
           $S[i].Gama\_a[1] = C_{L_{req}^{\max}}^{1} + t_{rt}^{\min} + C_{L_{resp}^{\max}}^{1} + t_{ID1m}^{1} + t_{ID1+}^{1} + t_{srLreq}^{1\to 2} + t_{rd}$
           $S[i].Gama\_b[1] = \max\left\{C_{L_{req}^{\max}}^{1} + t_{rt}^{\min} + t_{srL_{resp}^{\max}}^{1\to 2} + t_{rd}, t_{srL_{req}^{\max}}^{1\to 2} + t_{rd} + C_{L_{req}^{\max}}^{2} + t_{ID1m}^{2}\right\} +$
                      $+ C_{L_{req}^{\max}}^{2} + t_{ID1m}^{2}$
           /* q_Gama[1] is always 0 due to the idle time */
           /* but we compute it next anyway */
           S[i].q_Gama[1] = max {S[i].Gama_b[1]-S[i].Gama_a[1],0}
           /* Computation of Gama for the other IS */
           For j = 2 to (S[i].ndp - 1)
                $S[i].Gama\_a[j] = \max\{S[i].Gama\_a[j-1], S[i].Gama\_b[j-1]\} +$
                            $+ t_{srLreq}^{j\to j+1} + t_{rd}$
```

```
58:                $S[i].Gama\_b[j] = \max\left\{ \begin{array}{l} S[i].Gama\_b[j-1] - C^{j}_{L^{\max}_{resp}} - t^{j}_{ID1m} + t^{j\to j+1}_{srL^{\max}_{resp}} + t_{rd}, \\ \sum_{l=1}^{j}\left(t^{l\to l+1}_{srL^{\max}_{req}} + t_{rd}\right) + C^{j+1}_{L^{\max}_{req}} + t^{j+1}_{ID1m} \end{array} \right\} +$
                   $+ C^{j+1}_{L^{\max}_{resp}} + t^{j+1}_{ID1m}$
59:                S[i].q_Gama[j] = max {S[i].Gama_b[j]-
                                                 -S[i].Gama_a[j],0}
60:             endfor /* j */
61:             $S[i]Qgama = \sum_{j=1}^{ndp-1}(S[i].q\_Gama[j])$
62:             /* Computation of Qfi */
63:             /* Computation of Fi for the first IS */
64:             $S[i].Fi\_a[1] = C^{1}_{L^{\max}_{req}} + t^{1}_{ID2m} + t^{1}_{ID2+} + t^{1\to 2}_{srLreq} + t_{rd}$
65:             $S[i].Fi\_b[1] = t^{1\to 2}_{srL^{\max}_{req}} + t_{rd} + C^{2}_{L^{\max}_{req}} + t^{2}_{ID2m}$
66:             /* q_Fi[1] is always 0 due to the idle time */
67:             /* but we compute it next anyway */
68:             S[i].q_Fi[1] = max {S[i].Fi_b[1]-S[i].Fi_a[1],0}
69:             /* Computation of Fi for the other IS */
70:             For j = 2 to (S[i].ndp - 1)
71:                $S[i].Fi\_a[j] = \max\{S[i].Fi\_a[j-1], S[i].Fi\_b[j-1]\} + t^{j\to j+1}_{srLreq} + t_{rd}$
72:                $S[i].Fi\_b[j] = \sum_{l=1}^{j}\left(t^{l\to l+1}_{srL^{\max}_{req}} + t_{rd}\right) + C^{j+1}_{L^{\max}_{req}} + t^{j+1}_{ID2m}$
73:                S[i].q_Fi[j] = max {S[i].Fi_b[j]-S[i].Fi_a[j],0}
74:             endfor /* j */
75:             $S[i]Qfi = \sum_{j=1}^{ndp-1}(S[i].q\_Fi[j])$
76:             /* Computation of Q (maximum total queuing) */
77:             S[i].Q = max {S[i].Qgama,S[i].Qfi}
78:          endif /* S[i].ndp = 2 */
79:      endif /* S[i].ndp = 1 */
80:      /* Computation of the total system turnaround time */
81:      S[i].tst = S[i].tstn + S[i].Q
82:      /* Computation of the duration of the message transaction */
83:      $S[i].duration = C^{1}_{Lreq} + t_{st} + C^{1}_{Lresp} + t^{1}_{ID1m} + t^{1}_{ID1+}$
84:  endfor /* For i */
85:  /* Computation of TSL1 */
86:  $t_{SL1}$ = max {S[i].tst}
87:  end.
```

D.2 Computation of T_{SL2}

The following algorithm (presented in pseudo-code) for the computation of T_{SL2} is proposed:

Algorithm D.2: Computation of TSL2

```
Begin
/* Define the Master structure type */
Master_type = record
          ndp: integer    /* number of Comm. Domains in the path */
                          /* between initiator and responder */
          path: array[10] of integer /* physical media in the path */
          Gama_a: array[10] of real
          Gama_b: array[10] of real
          q_Gama: array[10] of real /* queuing Gama in each domain */
          Qgama: real     /* total queuing Gama in the path */
          Fi_a: array[10] of real
          Fi_b: array[10] of real
          q_Fi: array[10] of real /* queuing Fi in each domain */
          QFi: real       /* total queuing Fi in the path */
          Q: real         /* total queuing in the path */
          tst_token: real /* system turnaround time for TSL2 */
          endrecord
/* Define the Master structure variable */
M: array[100] of Master_type
/* Call the Idle_Time procedure */
Call Idle_Time
/* Read Stream-related parameters */
input(nmasters)           /* number of masters */
For i=1 to nmasters       /* For every Stream */
    input(M[i].ndp)
    for j=1 to M[i].ndp
        input(M[i].path[j])
    endfor /* j */
endfor /* i */
/* Computation of the worst-case system turnaround after token */
For i=1 to nmasters       /* For every master */
    if M[i].ndp = 1 then  /* if no IS in the path */
        /* tst_token = $T_{ID1}$ for the physical medium of that master */
        M[i].tst_token = $t^{i}_{ID1m} + t^{i}_{ID1+}$
    else                  /* if one or more IS in the path */
```

```
/* Computation of the queuing delays affecting the token */
if M[i].ndp = 2 then  /* only one IS in the path */
   M[i].Q = 0         /* no queuing in first IS */
else                  /* two or more IS in the path */
   /* Computation of Qgama */
   /* Computation of Gama for the first IS */
   $M[i].Gama\_a[1] = C^{1}_{L^{\max}_{req}} + t^{\min}_{rt} + C^{1}_{L^{\max}_{resp}} + t^{1}_{ID1m} + t^{1}_{ID1+} + t^{1\to 2}_{srLtoken} + t_{rd}$
   $M[i].Gama\_b[1] = \max\left\{C^{1}_{L^{\max}_{req}} + t^{\min}_{rt} + t^{1\to 2}_{srL^{\max}_{resp}} + t_{rd}, t^{1\to 2}_{srL^{\max}_{req}} + t_{rd} + C^{2}_{L^{\max}_{req}} + t^{2}_{ID1m}\right\} +$
            $+ C^{2}_{L^{\max}_{req}} + t^{2}_{ID1m}$
   /* q_Gama[1] is always 0 due to the idle time */
   /* but we compute it next anyway */
   M[i].q_Gama[1] = max {M[i].Gama_b[1]-M[i].Gama_a[1],0}
   /* Computation of Gama for the other IS */
   For j = 2 to (M[i].ndp - 1)
      $M[i].Gama\_a[j] = \max\{M[i].Gama\_a[j-1], M[i].Gama\_b[j-1]\} +$
            $+ t^{j\to j+1}_{srLtoken} + t_{rd}$
      $M[i].Gama\_b[j] = \max\left\{\begin{matrix} M[i].Gama\_b[j-1] - C^{j}_{L^{\max}_{resp}} - t^{j}_{ID1m} + t^{j\to j+1}_{srL^{\max}_{resp}} + t_{rd}, \\ \sum_{l=1}^{j}\left(t^{l\to l+1}_{srL^{\max}_{req}} + t_{rd}\right) + C^{j+1}_{L^{\max}_{req}} + t^{j+1}_{ID1m} \end{matrix}\right\} +$
            $+ C^{j+1}_{L^{\max}_{resp}} + t^{j+1}_{ID1m}$
      M[i].q_Gama[j] = max {M[i].Gama_b[j]-
                      - M[i].Gama_a[j],0}
   endfor /* j */
   $M[i].Qgama = \sum_{j=1}^{ndp-1}(M[i].q\_Gama[j])$
   /* Computation of Qfi */
   /* Computation of Fi for the first IS */
   $M[i].Fi\_a[1] = C^{1}_{L^{\max}_{req}} + t^{1}_{ID2m} + t^{1}_{ID2+} + t^{1\to 2}_{srLtoken} + t_{rd}$
   $M[i].Fi\_b[1] = t^{1\to 2}_{srL^{\max}_{req}} + t_{rd} + C^{2}_{L^{\max}_{req}} + t^{2}_{ID2m}$
   /* q_Fi[1] is always 0 due to the idle time */
   /* but we compute it next anyway */
   M[i].q_Fi[1] = max {M[i].Fi_b[1]-M[i].Fi_a[1],0}
   /* Computation of Fi for the other IS */
   For j = 2 to (M[i].ndp - 1)
      $M[i].Fi\_a[j] = \max\{M[i].Fi\_a[j-1], M[i].Fi\_b[j-1]\} + t^{j\to j+1}_{srLtoken} + t_{rd}$
      $M[i].Fi\_b[j] = \sum_{l=1}^{j}\left(t^{l\to l+1}_{srL^{\max}_{req}} + t_{rd}\right) + C^{j+1}_{L^{\max}_{req}} + t^{j+1}_{ID2m}$
```

```
65:                 M[i].q_Fi[j] = max {M[i].Fi_b[j]-M[i].Fi_a[j],0}
66:             endfor /* j */
67:             $M[i].Qfi = \sum_{j=1}^{ndp-1}(M[i].q\_Fi[j])$
68:             /* Computation of Q (maximum total queuing) */
69:             M[i].Q = max {M[i].Qgama,M[i].Qfi}
70:          endif /* M[i].ndp = 2 */
71:      endif /* M[i].ndp = 1 */
72:      /* Computation of the sum of $t_{srLtoken}$ */
73:      $Sum\_t_{srLtoken} = \sum_{j=1}^{ndp-1}\left(t_{srL_{token}}^{j\to j+1} + t_{rd}\right)$
74:      /* Computation of the maximum sum of the $t_{sr}$ */
75:      /* of the following master's request or token PDU */
76:      /* Considering that the following master issues a request PDU */
77:      For $L_{req}$ = $L^{min}{}_{req}$ to $L^{max}{}_{req}$ /* for all possible request lengths */
78:          $Sum\_t_{sr}(Lreq) = \sum_{j=ndp}^{2}\left(t_{srL_{req}}^{j\to j-1} + t_{rd}\right)$
79:      endfor /* j */
80:      $MaxSum\_t_{srLreq} = \max(Sum\_t_{sr}(Lreq))$
81:      /* Considering that the following master issues a token PDU */
82:      /* of the following master's request or token PDU */
83:      $Sum\_t_{srLtoken} = \sum_{j=ndp}^{2}\left(t_{srL_{token}}^{j\to j-1} + t_{rd}\right)$
84:      /* The worst-case is the maximum between the two */
85:      $MaxSum\_t_{sr} = \max(MaxSum\_t_{srLreq}, Sum\_t_{srLtoken})$
86:      /* The system turnaround time after a token can be computed */
87:      $M[i].tst\_token = M[i].Q + Sum\_t_{srLtoken} + C_{Ltoken}^{ndp} + t_{ID1m}^{ndp} + t_{ID1+}^{ndp} +$
             $+ MaxSum\_t_{sr} - C_{Ltoken}^{1}$
88:  endfor /* For i */
89:  /* Computation of TSL2 */
90:  $t_{SL2}$ = max {M[i].tst_token}
91:  end.
```

Annex E

Algorithm for the Computation of the Mobility Management Parameters

This annex presents an algorithm (presented in pseudo-code) intended for the computation of both the T_{ID2} for the Mobility Master and of the number of beacons (n_b) that each SIS/SLIS must transmit, during the mobility management period.

E.1 Simplified Algorithm for the Computation of T_{ID2} and n_b

Algorithm E.1: Computation of Mobility Management parameters

```
Begin
/* Define the Beacon Tigger (BT) structure type */
BT_type = record
          ndp: integer    /* number of Comm. Domains in the path */
                          /* between MobM and SWLD */
          path: array[10] of integer /* physical media in the path */
          tbtn: real      /* minimum latency of the BT PDU (no Q) */
          Gama_a: array[10] of real
          Gama_b: array[10] of real
          q_Gama: array[10] of real /* queuing Gama in each domain */
          Qgama: real     /* total queuing Gama in the path */
          Fi_a: array[10] of real
          Fi_b: array[10] of real
          q_Fi: array[10] of real /* queuing Fi in each domain */
          QFi: real       /* total queuing Fi in the path */
          Delta_a: array[10] of real
          Delta_b: array[10] of real
          q_Delta: array[10] of real /* queuing Fi in each domain */
          QDelta: real    /* total queuing Fi in the path */
          Q: real         /* total queuing in the path */
          tbt: real /* worst-case latency of the BT PDU */
          tbp_pre: real /* prel. duration of the beacon period  */
          tbp: real /* duration of the beacon period  */
          tbp: real /* duration of the beacon period  */
          nb: longint /* number of beacons  */
          endrecord
Mob_type = record
          BT: array[10] of BT_type /* one for each SIS/SLIS */
          Lbt: integer /* length of the BT PDU (chars) */
          tbgap: real /* duration of the beacon gap (us) */
          tsw: real /* duration of the channel switching (us) */
          Cbeacon: real /* duration of the beacon (us) */
          nch: integer /* number of radio channel sets */
          nSIS: integer /* number of SIS/SLIS */
          tbt: real /* max latency of the BT PDU, for all SWLD */
          tho: real /* max duration of the handoff procedure */
          tmob_pre: real /* prel. duration of mobility management */
          tmob: real /* final duration of mobility management */
          tID2plus: real /* inserted idle time after the BT PDU */
          TID2: longint /* Idle Time parameter in the MobM */
          endrecord
/* Define the MobM structure variable */
Mob: Mob_type
/* MobM is exclusively dedicated to mobility or a normal master? */
MobM_dedicated: integer /* 0 – normal master, 1 – dedicated master */
/* Call the Idle_Time procedure */
Call Idle_Time
/* Read Mobility-related parameters */
input(MobM_dedicated)
input(Mob.nch)
input(Mob.Lbt)
input(Mob.tbgap)
input(Mob.tsw)
input(Mob.Cbeacon)
input(Mob.nSIS)
For i=1 to Mob.nSIS        /* For every SIS/SLIS */
    input(Mob.BT[i].ndp)
    for j=1 to Mob.BT[i].ndp
        input(Mob.BT[i].path[j])
    endfor /* j */
endfor /* i */
/* Computation of the Mobility Management parameters */
For i=1 to Mob.nSIS        /* For every SIS/SLIS */
    /* Computation of the queuing delays affecting the BT PDU */
    /* There is at least one SIS/SLIS between the MobM and a SWLD */
    if Mob.BT[i].ndp = 2 then    /* only one IS in the path */
        Mob.BT[i].Q = 0          /* no queuing delays */
    else                    /* two or more IS in the path */
        if MobM_dedicated = 0 then /* MobM is a normal master */
            /* In this case, the BT PDU is assumed to be preceded */
            /* by maximum length ack. or unack. transactions */
            /* Computation of Qgama */
            /* Computation of Gama for the first IS */
            $Mob.BT[i].Gama\_a[1] = C^{1}_{L^{\max}_{req}} + t_{rt} + C^{1}_{L^{\max}_{resp}} + t^{1}_{ID1m} + t^{1}_{ID1+} + t^{1\to 2}_{srBT} + t_{rd}$
            $Mob.BT[i].Gama\_b[1] = \max\left\{C^{1}_{L^{\max}_{req}} + t_{rt} + t^{1\to 2}_{srL^{\max}_{resp}} + t_{rd}, t^{1\to 2}_{srL^{\max}_{req}} + t_{rd} + C^{2}_{L^{\max}_{req}} + t^{2}_{ID1m}\right\} +$
                        $+ C^{2}_{L^{\max}_{req}} + t^{2}_{ID1m}$
            /* q_Gama[1] is always 0 due to the idle time */
            /* but we compute it next anyway */
            Mob.BT[i].q_Gama[1]= max{Mob.BT[i].Gama_b[1] -
                                     - M[i].Gama_a[1],0}
            /* Computation of Gama for the other IS */
            For j = 2 to (Mob.BT[i].ndp - 1)
                $Mob.BT[i].Gama\_a[j] = \max\left\{\begin{matrix} Mob.BT[i].Gama\_a[j-1], \\ Mob.BT[i].Gama\_b[j-1] \end{matrix}\right\} + t^{j\to j+1}_{srBT} + t_{rd}$
                $Mob.BT[i].Gama\_b[j] =$
                $\max\left\{\begin{matrix} Mob.BT[i].Gama\_b[j-1] - C^{j}_{L^{\max}_{resp}} - t^{j}_{ID1m} + t^{j\to j+1}_{srL^{\max}_{resp}} + t_{rd}, \\ \sum_{l=1}^{j}\left(t^{l\to l+1}_{srL^{\max}_{req}} + t_{rd}\right) + C^{j+1}_{L^{\max}_{req}} + t^{j+1}_{ID1m} \end{matrix}\right\} + C^{j+1}_{L^{\max}_{resp}} + t^{j+1}_{ID1m}$
              Mob.BT[i].q_Gama[j] = max {Mob.BT[i].Gama_b[j]-
                                         - Mob.BT[i].Gama_a[j],0}
            endfor /* j */
            $Mob.BT[i].Qgama = \sum_{j=1}^{ndp-1}\left(Mob.BT[i].q\_Gama[j]\right)$
            /* Computation of Qfi */
            /* Computation of Fi for the first IS */
            $Mob.BT[i].Fi\_a[1] = C^{1}_{L^{\max}_{req}} + t^{1}_{ID2m} + t^{1}_{ID2+} + t^{1\to 2}_{srBT} + t_{rd}$
            $Mob.BT[i].Fi\_b[1] = t^{1\to 2}_{srL^{\max}_{req}} + t_{rd} + C^{2}_{L^{\max}_{req}} + t^{2}_{ID2m}$
            /* q_Fi[1] is always 0 due to the idle time */
            /* but we compute it next anyway */
            Mob.BT[i].q_Fi[1] = max { Mob.BT[i].Fi_b[1] -
                                    - Mob.BT[i].Fi_a[1],0}
            /* Computation of Fi for the other IS */
            For j = 2 to (Mob.BT[i].ndp - 1)
                $Mob.BT[i].Fi\_a[j] = \max\left\{\begin{matrix} Mob.BT[i].Fi\_a[j-1], \\ Mob.BT[i].Fi\_b[j-1] \end{matrix}\right\} + t^{j\to j+1}_{srBT} + t_{rd}$
                $Mob.BT[i].Fi\_b[j]=\sum_{l=1}^{j}\left(t_{srL_{req}^{\max}}^{l\rightarrow l+1}+t_{rd}\right)+C_{L_{req}^{\max}}^{j+1}+t_{ID2m}^{j+1}$
            Mob.BT[i].q_Fi[j] = max { Mob.BT [i].Fi_b[j]-
                                      - Mob.BT[i].Fi_a[j],0}
        endfor /* j */
        $Mob.BT[i].Qfi=\sum_{j=1}^{ndp-1}\left(Mob.BT[i].q\_Fi[j]\right)$
        /* Computation of Q (maximum total queuing) */
        Mob.BT[i].Q = max {Mob.BT[i].Qgama,Mob.BT[i].Qfi}
    else /* MobM_dedicated = 1 */
        /* In this case, the BT PDU is always preceded */
        /* by a token PDU */
        /* Computation of Qdelta */
        /* Computation of Delta for the first IS */
        $Mob.BT[i].Delta\_a[1]=C_{L_{token}}^{1}+t_{ID1m}^{1}+t_{ID1+}^{1}+t_{srBT}^{1\rightarrow 2}+t_{rd}$
        $Mob.BT[i].Fi\_b[1]=t_{srL_{token}}^{1\rightarrow 2}+t_{rd}+C_{L_{token}}^{2}+t_{ID1m}^{2}$
        /* q_Delta[1] is always 0 due to the idle time */
        /* but we compute it next anyway */
        Mob.BT[i].q_Delta[1] = max { Mob.BT[i].Delta_b[1] -
                                 - Mob.BT[i].Delta_a[1],0}
        /* Computation of Delta for the other IS */
        For j = 2 to (Mob.BT[i].ndp - 1)
            $Mob.BT[i].Delta\_a[j]=\max\left\{\begin{matrix}Mob.BT[i].Delta\_a[j-1],\\ Mob.BT[i].Delta\_b[j-1]\end{matrix}\right\}+t_{srBT}^{j\rightarrow j+1}+t_{rd}$
            $Mob.BT[i].Delta\_b[j]=\sum_{l=1}^{j}\left(t_{srL_{token}}^{l\rightarrow l+1}+t_{rd}\right)+C_{L_{token}}^{j+1}+t_{ID1m}^{j+1}$
            Mob.BT[i].q_Delta[j] = max { Mob.BT[i].Delta_b[j]-
                                      - Mob.BT[i].Delta_a[j],0}
        endfor /* j */
        $Mob.BT[i].Qdelta=\sum_{j=1}^{ndp-1}\left(Mob.BT[i].q\_Delta[j]\right)$
        /* Computation of Q (maximum total queuing) */
        Mob.BT[i].Q = Mob.BT[i].Qdelta
    endif /* MobM_dedicated */
endif /* Mob.BT[i].ndp */
/* Computation of the sum of $t_{srBT}$ */
$Sum\_t_{srBT}=\sum_{j=1}^{ndp-1}\left(t_{srL_{BT}}^{j\rightarrow j+1}+t_{rd}\right)$
/* Computation of the latency of the BT PDU without queuing */
```

```
    $Mob.BT[i].tbtn = Sum\_t_{srBT} + C_{BT}^{ndp} - C_{BT}^{1}$
    /* Computation of the latency of the BT PDU */
    $Mob.BT[i].tbt = Mob.BT[i].Q + Mob.BT[i].tbtn$
endfor /* i */
/* Get the maximum latency for the BT PDU, for all SWLD */
Mob.tbt = max {Mob.BT[i].tbt}
/* Computation of the maximum duration of the handoff procedure */
Mob.tho = (2*Mob.nch-1)*Mob.Cbeacon + Mob.nch*(Mob.tbgap+Mob.tsw)
/* Compute the preliminary duration of the beacon period */
Mob.tmob_pre = Mob.tbt + Mob.tho
/* Compute the number of beacons for every SIS/SLIS */
For i=1 to Mob.nSIS       /* For every SIS/SLIS */
    /* Compute the maximum duration of the beacon period */
    /* in that particular SIS */
    Mob.BT[i].tbp_pre = Mob.tmob_pre - Mob.BT[i].tbtn
    /* Compute the number of beacons that must be transmitted */
    Mob.BT[i].nb = ceil(Mob.BT[i].tbp_pre/(Mob.tbgap+Mob.tsw))
    /* Recompute the beacon period duration */
    Mob.BT[i].tbp = Mob.BT[i].nb*(Mob.tbgap+Mob.tsw))
    /* Recompute the mobility management duration */
    Mob.BT[i].tmob = Mob.BT[i].tbt + Mob.BT[i].tbp
endfor /* i */
/* Compute the final mobility management duration */
/* that is the maximum between all */
Mob.tmob = max {Mob.BT[i].tmob}
/* Compute the Idle Time parameter */
Mob.TID2 = ceil(Mob.tmob * r(MobM))
Mob.tID2plus = Mob.tmob – TIDm / r(MobM)
end.
```

Annex F

Input Data File for the System Planning Tool

This annex presents a text file that is used as input data for the program that computes all network parameters (e.g. T_{ID1}, T_{ID2}, t_{st} and C_{ack} for all message streams, T_{SL1}, T_{SL2} and the mobility management parameters).

F.1 Input data file

```
Communication-Network-Specific
255     Lreqmax - maximum length of the request DLL PDUs (characters)
255     Lresp_max - maximum length of the response DLL PDUs (characters)
6       Lreq_min - minimum length of the request DLL PDUs (characters)
6       Lresp_min - minimum length of the response DLL PDUs (characters)
3       Ltoken - length of the token DLL PDU (characters)
8       d - number of data bits of each DLL character (bits)
10      trt_min - minimum responder's turnaround time (us)
50      trt_max - maximum responder's turnaround time (us)
100     TIDm - minimum idle time (bits)
25      trd - relaying delay (us)

Physical-Media-Specific (one row for each physical media (nm rows))
2       nm - number of (different) physical media
r(Mbaud)       lH(bits)        lT(bits)        k(bits)         o(bits)
1.5             0               0               3               33
2               200             0               0               150

Message Streams
24      ns - number of message streams

S1
1       ndp - number of communication domains in the path
1       Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S2
1       ndp - number of communication domains in the path
1       Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)
```

```
S3
1       ndp - number of communication domains in the path
1       Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S4
2       ndp - number of communication domains in the path
1 2     Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S5
4       ndp - number of communication domains in the path
1 2 1 2    Path - physical medium of every communication domain in the
path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S6
2       ndp - number of communication domains in the path
1 2     Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)

S7
4       ndp - number of communication domains in the path
1 2 1 2    Path - physical medium of every communication domain in the
path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)

S8
2       ndp - number of communication domains in the path
1 2     Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S9
4       ndp - number of communication domains in the path
1 2 1 2    Path - physical medium of every communication domain in the
path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S10
3       ndp - number of communication domains in the path
1 2 1   Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S11
3       ndp - number of communication domains in the path
1 2 1   Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)
```

```
S12
3       ndp - number of communication domains in the path
1 2 1   Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S13
4       ndp - number of communication domains in the path
1 2 1 2 Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S14
4       ndp - number of communication domains in the path
1 2 1 2 Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)

S15
4       ndp - number of communication domains in the path
1 2 1 2 Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S16
2       ndp - number of communication domains in the path
2 1     Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S17
2       ndp - number of communication domains in the path
2 1     Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)

S18
2       ndp - number of communication domains in the path
2 1     Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S19
4       ndp - number of communication domains in the path
2 1 2 1 Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S20
2       ndp - number of communication domains in the path
2 1  Path - physical medium of every communication domain in the path
255     Lreq - Length of the request DLL PDU (characters)
6       Lresp - Length of the response DLL PDU (characters)

S21
4       ndp - number of communication domains in the path
2 1 2 1 Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)
```

```
S22
2       ndp - number of communication domains in the path
2 1  Path - physical medium of every communication domain in the path
59      Lreq - Length of the request DLL PDU (characters)
59      Lresp - Length of the response DLL PDU (characters)

S23
4       ndp - number of communication domains in the path
2 1 2 1 Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

S24
2       ndp - number of communication domains in the path
2 1  Path - physical medium of every communication domain in the path
6       Lreq - Length of the request DLL PDU (characters)
255     Lresp - Length of the response DLL PDU (characters)

Number of Token PDU (including all possible paths and including the MobM)
4       ntokens

Token 1
4       ndp - number of communication domains in the path
1 2 1 2 Path - Path of the token until next master

Token 2
2       ndp - number of communication domains in the path
1 2  Path - Path of the token until next master

Token 3
4       ndp - number of communication domains in the path
2 1 2 1 Path - Path of the token until next master

Token 4
2       ndp - number of communication domains in the path
2 1  Path - Path of the token until next master

Is there Mobility support?
1       Mobility (0 - MobM Absent, 1 - MobM Present)

Is the Mobility Master (MobM) exclusively dedicated to mobility
management?
0       MobM_exclusive (0 - MobM normal master, 1 - MobM dedicated master)

Parameters for the mobility management
3       nch - number of radio channel sets
10      Lbt - Length of the Beacon Trigger (BT) PDU (in chars)
25      tbgap - time interval between beacons (in us)
100     tsw - switching delay, from one radio channel to another (in us)
100     Cbeacon - duration of the beacon (us)

Number of SIS/SLIS
3       nSIS

Path of the Beacon Trigger (BT) PDU for SWLD1
2       ndp - number of communication domains in the path
1 2     Path - Path of the BT PDU between MobM and SWLD
```

```
Path of the Beacon Trigger (BT) PDU for SWLD2
4       ndp - number of communication domains in the path
1 2 1 2 Path - Path of the BT PDU between MobM and SWLD

Path of the Beacon Trigger (BT) PDU for SWLD3
4       ndp - number of communication domains in the path
1 2 1 2 Path - Path of the BT PDU between MobM and SWLD
```

Annex G

Acronyms and Symbols

This annex presents a description of acronyms and symbols that are used throughout this Thesis. Additionally, a reference to the section where the acronym/symbol is defined is also provided.

G.1. List of Acronyms

Table G.1: List of Acronyms

Acronym	Description	Section
10GEA	10 Gigabit Ethernet Alliance	2.2.3
AL	Application Layer	3.2.1
ARC	Ad-hoc Radio Cell	4.2.2
AWLD	Ad-hoc Wireless (Communication) Domain	4.2.2
BER	Bit Error Rate	2.2.4
BT	Beacon Trigger (PDU)	4.5.2
CENELEC	European Committee for Electrotechnical Standardisation	2.2.2
CH	Radio Channel	4.2.2
CHS	Radio Channel Set	4.2.2
COTS	Commercial Off-The-Shelf	2.2.5
CSRD	Cyclic Send and Request Data (PROFIBUS)	3.2.2
DA	Destination Address (PROFIBUS)	3.2.7
DCCS	Distributed Computer-Controlled System	2.1
DECT	Digital Enhanced Cordless Telecommunications	2.2.4
DLL	Data Link Layer (OSI model)	3.1
DLL PDU	Data Link Layer Protocol Data Unit	4.1
ED	End Delimiter (PROFIBUS)	3.2.7
ES	End System	4.2.1
FC	Frame Control (PROFIBUS)	3.2.7
FCS	Frame Check Sequence (PROFIBUS)	3.2.7
HIPERLAN	High Performance Radio Local Area Network	2.2.4
HSA	Highest Station Address (PROFIBUS)	3.2.6
HSE	High Speed Ethernet (Foundation Fieldbus)	2.2.3
I	Initiator	5.7.1
IAONA	Industrial Automation Networking Alliance	2.2.3
IEA	Industrial Ethernet Association	2.2.3
IS	Intermediate System	4.2.1
LAN	Local Area Network	2.1
LAS	List of Active Stations (PROFIBUS)	3.2.6
LE	LEngth field (PROFIBUS)	3.2.7
LEr	LEngth field Repeated (PROFIBUS)	3.2.7
LGAP	GAP List (PROFIBUS)	3.2.6
LIS	Linking Intermediate System	4.2.3

Acronym	Description	Section
MAC	Medium Access Control	1.1
MAP	Manufacturing Automation Protocol	2.2.1
MLIS	Mobile Linking Intermediate System	4.2.3
MMS	Manufacturing Message Specification	2.2.1
MobM	Mobility Master	4.5.2
MWRD	Mobile Wired (Communication) Domain	4.2.2
NS	Next Station (PROFIBUS)	3.2.4
OOA	Object-Oriented Analysis	A.1
PDU	Protocol Data Unit	4.1
PhL	Physical Layer (OSI model)	3.1
PhL PDU	Physical Layer Protocol Data Unit	4.1
PS	Previous Station address (PROFIBUS)	3.2.6
R	Responder	5.7.1
RC	Radio Cell	4.2.2
SA	Source Address (PROFIBUS)	3.2.7
SD	Start Delimiter (PROFIBUS)	3.2.7
SDA	Send Data with Acknowledge (PROFIBUS)	3.2.2
SDN	Send Data with No acknowledge (PROFIBUS)	3.2.2
SIS	Structuring Intermediate System	4.2.3
SLIS	Structuring and Linking Intermediate System	4.2.3
SRC	Structured Radio Cell	4.2.2
SRD	Send and Request Data (PROFIBUS)	3.2.2
SWLD	Structured Wireless (Communication) Domain	4.2.2
TS	This Station address (PROFIBUS)	3.2.4
UMTS	Universal Mobile Telecommunication System	2.2.4
WLAN	Wireless Local Area Network	2.1
WLD	Wireless (Communication) Domain	4.2.2
WLES	Wireless End System	4.2.1
WPAN	Personal Local Area Network	2.1
WRD	Wired (Communication) Domain	4.2.2
WRES	Wired End System	4.2.1

G.2. List of Symbols

Table G.2: List of Symbols

Symbol	Description	Section
$\Gamma^{i\to j}_a$	The time elapsed from the beginning of transmission of the acknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the request PDU of transaction l (req(l)) may start to be transmitted in D^j.	6.3
$\gamma^{i\to j}_a$	The time elapsed from the beginning of transmission of the acknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the response PDU of transaction l-1 (resp(l-1)) may start to be transmitted in D^j.	6.3
$\Delta^{i\to j}_a$	Time elapsed from the beginning of transmission of the token PDU in D^i, until the moment when the request PDU of transaction l (req(l)) may start to be transmitted in D^j.	6.4
$\Phi^{i\to j}_a$	Time elapsed from the beginning of transmission of the unacknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the request PDU of transaction l (req(l)) may start to be transmitted in D^j.	6.6
$\Gamma^{i\to j}_b$	The time elapsed from the beginning of transmission of the acknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the IS is able to start transmitting the request PDU of transaction l (req(l)) in D^j.	6.3

Symbol	Description	Section
$\gamma^{i\rightarrow j}_{b}$	The time elapsed from the beginning of transmission of the acknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the IS is able to start transmitting the response PDU of transaction l-1 (resp(l-1)) in D^j.	6.3
$\Delta^{i\rightarrow j}_{b}$	Time elapsed from the beginning of transmission of the token PDU in D^i, until the moment when the IS can start transmitting the request PDU of transaction l (req(l)) in D^j.	6.4
$\Phi^{i\rightarrow j}_{b}$	Time elapsed from the beginning of transmission of the unacknowledged request PDU of transaction l-1 (req(l-1)) in D^i, until the moment when the IS is able to start transmitting the request PDU of transaction l (req(l)) in D^j.	6.6
Γ^{i}_{a}	Adaptation of $\Gamma^{i\rightarrow j}_{a}$ to encompass more than one IS between initiator and responder.	7.3.4
Φ^{i}_{a}	Adaptation of $\Phi^{i\rightarrow j}_{a}$ to encompass more than one IS between initiator and responder.	7.3.5
Δ^{i}_{a}	Adaptation of $\Delta^{i\rightarrow j}_{a}$ to encompass more than one IS between initiator and responder.	8.4.2
Γ^{i}_{b}	Adaptation of $\Gamma^{i\rightarrow j}_{b}$ to encompass more than one IS between initiator and responder.	7.3.4
Φ^{i}_{b}	Adaptation of $\Phi^{i\rightarrow j}_{b}$ to encompass more than one IS between initiator and responder.	7.3.5
Δ^{i}_{b}	Adaptation of $\Delta^{i\rightarrow j}_{b}$ to encompass more than one IS between initiator and responder.	8.4.2
C^{1}_{BT}	Duration of the BT PDU in Communication Domain 1 (location of the MobM)	8.4.2
C_{ack}	Duration of an acknowledged (request/response) transaction	7.4.1
C_{beacon}	Duration of the beacon (in the network)	5.5.2
C^{i}	Duration of a PhL PDU in domain D^i	5.3.3
$C^{i}_{req(l)}$	Duration of the request PDU of transaction l in D^i	5.3.3
$C^{i}_{req(l-2)}$	Duration of the request PDU of transaction (l-2) in D^i	5.3.3
$C^{i}_{resp(l-2)}$	Duration of the response PDU of transaction (l-2) in D^i	5.3.3
$C^{j}_{req(l)}$	Duration of the request PDU of transaction l in D^j	5.3.3
$C^{j}_{req(l-2)}$	Duration of the request PDU of transaction (l-2) in D^j	5.3.3
$C^{j}_{resp(l-2)}$	Duration of the response PDU of transaction (l-2) in D^j	5.3.3
C^{ndp}_{BT}	Duration of the BT PDU in Communication Domain ndp (SWLD being considered)	8.4.2
C_{unk}	Duration of an unacknowledged transaction	7.4.2
D	Set of Communication Domains in the network	5.2.1
d	Number of bits per char	5.3.3
D_TYPE	Communication Domain's type; D_TYPE $\in \{$WRD, MWRD, AWLD, SWLD$\}$	5.2.2
D^i	(Communication) Domain i; $D^i \in$ D	5.2.2
ES	Set of End-Systems in the network	5.2.1
ES(D^i)	Function that returns the set of all ESs attached to D^i	5.2.2
ES_ROLE	End System's role; ES_ROLE $\in \{$MASTER, SLAVE, MobM$\}$	5.2.3
ES_TYPE	End System's type; ES_TYPE $\in \{$WRES, WLES, MWLES$\}$	5.2.3
ES^j	End System j; $ES^j \in$ ES	5.2.3
G	Gap Update Factor is a constant that must be set in every master ES; T_{GUD} is reset to a multiple of the Target Rotation Time ($T_{GUD} = G \times T_{TR}$) after a complete GAP check, which may last several token rotations	3.2.6
HSA	Highest Station Address (PROFIBUS)	3.2.6
INITIATOR	ES that is the initiator of the transaction INITIATOR $\in \{ES^1,\ldots, ES^{ne}\}$	5.5.1
IS	Set of Intermediate Systems in the network	5.2.1
IS(D^i)	Function that returns the set of all ISs attached to D^i	5.2.2
IS_TYPE	Intermediate System's type; IS_TYPE $\in \{$SIS, LIS, MLIS, SLIS$\}$	5.4.1

Symbol	Description	Section
IS^k	Intermediate System k; $IS^k \in$ IS	5.4.1
k^l	Overhead per char for the PhL protocol of physical medium l	5.3.1
L	Length of the DLL PDU	5.3.3
L_{BT}	Length of the Beacon Trigger (BT) PDU (in the network)	5.5.2
l^l_H	Overhead of the head per PhL PDU in physical medium l	5.3.1
l^l_T	Overhead of the tail per PhL PDU in physical medium l	5.3.1
L^{max}_{req}	Maximum length of a DLL request PDU (in the network)	5.5.2
L^{max}_{resp}	Maximum length of a DLL response PDU (in the network)	5.5.2
L^{min}_{req}	Minimum length of a DLL request PDU (in the network)	5.5.2
L^{min}_{resp}	Maximum length of a DLL response PDU (in the network)	5.5.2
L_{req}	Length of the DLL request PDU	5.5.1
$L_{req(l)}$	Length of the DLL request (acknowledged or unacknowledged) or token PDU for transaction l	6.3
$L_{req(l-1)}$	Length of the DLL request PDU for transaction (l-1)	6.3
L_{resp}	Length of the DLL response PDU	5.5.1
$L_{resp(l-1)}$	Length of the DLL response PDU for transaction (l-1)	6.3
L_{token}	Length of the token PDU (in the network)	5.5.2
M	Set of Physical Media in the network	5.2.1
$MaxSum\_t_{sr}$	Maximum time elapsed since the token was received until either a request or token PDU is detected, by the original token transmitter	7.5.3
m^k	Set of the two physical media IS^k interconnects; $m^k = \{M^{l1}, M^{l2}\}$	5.4.1
M^l	Physical medium l; $M^l \in$ M	5.2.2
n_b(SWLD)	Number of beacons that a particular SLIS (or SIS) must transmit	
nch	Number of radio channel sets in use (in the network)	5.5.2
nd	Number of Communication Domains in the network.	5.2.1
ndp	Number of Communication Domains in the path (between the initiator and the responder of a transaction).	7.3.4
ne	Number of End Systems in the network	5.2.1
ni	Number of Intermediate Systems in the network	5.2.1
nm	Number of Physical Media in the network	5.2.1
N^n	(Communication) Network n;	5.5.2
ns	Number of Message Streams in the network	5.2.1
NS	Next Station address (PROFIBUS)	3.2.6
o^l	offset defining the total number of bits until knowing the length of the data field, in physical medium l	5.3.2
path	Ordered set of Communication Domains between the initiator and the responder of a transaction.	7.3.4
PS	Previous Station address (PROFIBUS)	3.2.6
q	Queuing delay affecting a PDU in an Intermediate System	5.7.1
Q	Worst-case total queuing delay affecting a request PDU, from initiator to responder. Q must be set to the maximum between the worst-case total queuing delays imposed in case the precedent transactions are acknowledged (Q_Γ) or unacknowledged (Q_Φ), respectively	7.2
Q_Γ	Worst-case total queuing delay considering that the precedent transaction is acknowledged.	7.3.4
Q_Φ	Worst-case total queuing delay considering that the precedent transaction is unacknowledged.	7.3.5
Q_Δ	Worst-case total queuing delay considering that the precedent PDU is a token.	8.4.2
q^i_Γ	Queuing delay introduced from Communication Domain i to Communication Domain (i+1), considering that the precedent transaction is acknowledged	7.3.4
q^i_Φ	Queuing delay introduced from Communication Domain i to Communication Domain (i+1), considering that the precedent transaction is unacknowledged	7.3.5
q^i_Δ	Queuing delay introduced from Communication Domain i to Communication Domain (i+1) considering that the precedent PDU is a token	8.4.2
RESPONDER	ES that is the responder of the transaction RESPONDER $\in \{ES^1, \ldots, ES^{ne}\}$	5.5.1

Symbol	Description	Section
r^l	Bit rate in physical medium l	5.3.1
S^s	Message Stream s	5.5.1
Sum_t_{srBT}	Total latency of the BT PDU along the path from the MobM until a certain SWLD	8.4.2
$Sum\_t_{srLtoken}$	Time for relaying the token PDU from the domain of the transmitter to the domain of the responder	7.5.3
t'_{bp}(SWLD)	Preliminary (maximum) duration of the beacon period, for a SWLD	
t'_{mob}	Preliminary value for the mobility management duration	8.4
$t^{1\rightarrow n}_{stn}$	System turnaround time of a transaction (l) between an initiator in D^1 and a responder in D^n, assuming no queuing delay.	7.2
t_{bgap}	The <u>b</u>eacon gap is the time interval between beacons (in the network)	5.5.2
t_{bp}(SWLD)	beacon period duration for a SWLD	
t_{bt}	Worst-case latency of the beacon trigger (BT) PDU	8.4.1
t_{btn}	Latency of the BT PDU from the Communication Domain of the MobM to the SWLD being considered, without considering queuing delays	8.4.2
t_{btn}(SWLD)	Worst-case latency of the beacon trigger (BT) PDU, assuming no queuing delays	
t^{curr}_{ass}	The time spent in the assessment of the radio channel currently being used by a mobile ES/LIS	8.4.3
T_{GUD}	<u>G</u>ap <u>Up</u>date Time is a time set in each master ES for updating the GAP List (PROFIBUS)	3.2.6
t_{ho}	Worst-case duration of the handoff procedure	8.4.1
$t^{i\rightarrow j}_{ID\Delta 1+}$	Additional idle time that would have to be inserted by a master ES in D^i, if the previous PDU was always a token PDU, considering the other domain as D^j.	6.4
$t^{i\rightarrow j}_{ID1\Gamma+}$	Additional idle time that would have to be inserted by a master ES in D^i, if the previous PDU was always a response PDU, considering the other domain as D^j.	6.3
$t^{i\rightarrow j}_{ID1+}$	The maximum between $t^{i\rightarrow j}_{ID1\Delta+}$ and $t^{i\rightarrow j}_{ID1\Gamma+}$	6.5
$t^{i\rightarrow j}_{ID2+}$	Additional idle time that would have to be inserted by a master ES in D^i, if the previous PDU was always an unacknowledged request PDU, considering the other domain as D^j.	6.6
$t^{i\rightarrow j}_{ng}$	The <u>n</u>o <u>g</u>aps instant is the earliest instant to start relaying the PhL PDU from D^i to D^j in a way that guarantees that the transmission in D^j is continuous.	5.4.3
$t^{i\rightarrow j}_{sr}$	The <u>s</u>tart <u>r</u>elaying instant is the earliest time instant for an IS to start relaying a specific PhL PDU from D^i to D^j, counted since the beginning of the PhL PDU in D^i.	5.4.3
$t^{i\rightarrow j}_{srreq(l-2)}$	Start relaying instant of the request PDU of transaction (l-2) from D^i to D^j	6.2
$t^{i\rightarrow j}_{srresp(l-1)}$	Start relaying instant of the response PDU of transaction (l-1) from D^i to D^j	6.2
T_{ID1}	Idle time inserted by a master ES after an acknowledgement, response or token PDU (PROFIBUS).	3.2.9
T_{ID1m}	Minimum value for T_{ID1}	6.2
T_{ID2}	Idle time inserted by a master ES after an acknowledged request PDU (PROFIBUS).	3.2.9
T_{ID2m}	Minimum value for T_{ID2}	6.2
T_{IDm}	Minimum inactivity (IDle) time introduced by the ESs/ISs between any two consecutive PhL PDUs (in the network)	5.5.2
t^i_{dr}	The <u>d</u>ata <u>r</u>eady instant is the instant at which a predefined amount of DLL data has been received from D^i (ready to be relayed), counted since the beginning of the PhL PDU in D^i.	5.4.3
$t^i_{ID1\Gamma+}$	The maximum of $t^{i\rightarrow j}_{ID1\Gamma+}$, considering all possible j.	6.3
$t^i_{ID1\Delta+}$	The maximum of $t^{i\rightarrow j}_{ID1\Delta+}$, considering all possible j.	6.4
t^i_{ID1+}	The maximum of $t^{i\rightarrow j}_{ID1+}$, considering all possible j.	6.5
T^i_{ID1+}	Additional idle time that must be inserted after receiving a response or token PDU, for a master ES in D^i	6.5
T^i_{ID2+}	Additional idle time that must be inserted after transmitting an unacknowledged request PDU, for a master ES in D^i	6.7

Symbol	Description	Section
t^{i}_{ID2+}	The maximum of $t^{i\to j}_{ID2+}$, considering all possible j.	6.6
t^{i}_{lk}	The length known instant is the instant at which the length of the DLL PDU in D^i is known, counted since the beginning of the PhL PDU in D^i.	5.4.3
$t^{j\to i}_{srresp(l)}$	Start relaying instant of the response PDU of transaction l from D^j to D^i	6.2
$t^{j\to j+1}_{srBT}$	Start relaying instant of the BT PDU, from Communication Domain j to Communication Domain j+1	8.4.2
T^{k}_{IDm}	Minimum idle (inactivity) time introduced by IS^k between any two consecutive PhL PDUs.	5.4.1
t^{k}_{rd}	Internal relaying delay of IS^k	5.4.1
t^{max}_{ass}	Maximum duration for the assessment of a Radio Channel	8.4.3
t^{max}_{rt}	Maximum responder’s turnaround time (in the network)	5.5.2
t^{min}_{rt}	Minimum responder’s turnaround time (in the network)	5.5.2
t_{mob}(SWLD)	Worst-case mobility management duration due to a SWLD	
t_{rd}	Internal relaying delay of the ISs (in the network)	5.5.2
T_{RR}	Real Rotation Time is the time between two consecutive token receptions	3.2.5
t_{rt}	The responder’s turnaround time is the time elapsed since a Responder ends receiving a request PDU, until it starts transmitting the correspondent response PDU.	3.2.9
TS	This Station address (PROFIBUS)	3.2.6
T_{SDI}	Station Delay of the Initiator (PROFIBUS)	3.2.9
T_{SDR}	Station Delay of a Responder (PROFIBUS)	3.2.9
T_{SL}	The Slot Time is a parameter set in every master ES that defines a timeout for listening for activity in the bus, after having transmitted an acknowledged request or token PDU. T_{SL} is set to the maximum between T_{SL1} and T_{SL2}.	3.2.10
T_{SL1}	Maximum time the initiator waits for the complete reception of the first frame character of the acknowledgement/response frame, after transmitting the last bit of the request frame	3.2.10
T_{SL2}	Maximum time the initiator waits after having transmitted the last bit of the token PDU until it detects the first bit of a PDU (either a request or the token) transmitted by the ES that received the token	3.2.10
T_{SM}	Safety margin (PROFIBUS)	3.2.9
t_{st}	The system turnaround time for a message transaction is the time elapsed since an Initiator ends transmitting a request PDU until it starts receiving the correspondent response PDU.	5.7.1
t_{st}	System turnaround time of a transaction (l) between an initiator in D^1 and a responder in D^n, assuming no queuing delay.	7.2
t_{st_token}	System turnaround time after transmitting the token PDU	7.5.3
t_{sw}	Switching delay between radio channels (in the network)	5.5.2
T_{SYN}	Synchronisation period of (at least) 33 idle bit periods (PROFIBUS PhL v1)	3.2.7
T_{TD}	Transmission Delay (PROFIBUS) is the propagation delay in the bus.	3.2.10
T_{TH}	Token Holding time (PROFIBUS) is the period during which a master station (holding the token) is allowed to perform message cycles	3.2.5
T_{TR}	Target Rotation Time is the expected time for a token cycle	3.2.5