Universidade do Porto
Faculdade de Engenharia
FEUP

A Multiple Logical Ring Approach to Real-time Wireless-enabled PROFIBUS Networks

By

Luís Miguel Moreira Lino Ferreira

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

November 2005

Doctoral Committee:

Prof. José MENDONÇA	Chairman
Prof. Hans HANSSON	External Examiners
Prof. Luís ALMEIDA	
Prof. Mário LEITÃO	Internal Examiners
Prof. Adriano CARVALHO	
Prof. Eduardo TOVAR	Supervisor
Prof. Francisco VASQUES	Co-Supervisor

A Multiple Logical Ring Approach to Real-time Wireless-enabled PROFIBUS Networks

Abstract

Fieldbus communication systems have become a common solution to the problem of interconnecting sensors, actuators and control devices in manufacturing automation and process control applications. Recently, there has been an enormous eagerness to extend fieldbus functionalities to support wired and wireless network stations in the same network. This thesis addresses the proposal of a novel architecture for a hybrid fieldbus communication system, where wired and wireless transmission media coexist.

The RFieldbus European project was one major effort towards a hybrid wired/wireless fieldbus solution. Although some of the achievements could potentially be applied to other commercial-off-the-shelf (COTS) standard fieldbuses, most of the effort in that project was devoted to an actual implementation over PROFIBUS (acronym for PROcess FIeld BUS) technologies. In our opinion, the arguments that were put forward in favour of using PROFIBUS as the federating communication system for such architecture are still valid.

In RFieldbus, the interconnection of wired and wireless stations is based on interconnecting devices operating at the Physical Layer level (as repeaters). In this thesis, we propose an alternative approach where the interconnecting devices act as bridges, thus operating at the Data Link Layer level.

The hypothesis is that such a bridge-based approach is devisable and presents, from the timeliness and reliability perspectives, advantages over the RFieldbus approach.

One of the contributions of this thesis is the specification of the Inter-Domain Protocol (IDP) which enables the execution of transactions between stations belonging to different media, i.e. transactions that must be relayed through one or more bridges. The IDP specifies the behaviour of the bridges when processing such kind of transactions, how the response can be obtained from the responder station (attached to another medium) and the format of the frames exchanged between bridges. The IDP builds upon the operational characteristics of the PROFIBUS-DP Application Layer, therefore guaranteeing full compatibility with this protocol.

In the proposed architecture, wireless stations can move between different wireless cells. In order to support this functionality, this thesis also proposes the Inter-Domain Mobility Procedure (IDMP). This protocol includes several operational phases, with the objective of being fully transparent to the system applications by guaranteeing no communication errors and no order inversion of frames. Therefore, the IDMP is compatible with standard PROFIBUS stations.

The IDP and the IDMP lead to additional communication delays, in relation to a standard PROFIBUS network, due to periods of network inaccessibility and the intrinsic operation of the IDP. Therefore, a timing analysis of the IDP is proposed. This analysis is then extended for integrating the effects of the IDMP on message transactions. These methodologies constitute a tool enabling the support of real-time applications. The thesis also shows the advantages of the proposed architecture over the RFieldbus approach, namely in terms of responsiveness to network errors, in terms of fault containment and it terms of timeliness for message transactions within the same network domain.

Keywords: Real-time systems; Real-time communications; Fieldbus networks; PROFIBUS.

Uma Abordagem Múltiplo Anel Lógico para Redes de Tempo-Real PROFIBUS com Extensões Rádio

Resumo

As redes de comunicação industrial do tipo redes de campo (*fieldbus*, em inglês) são, hoje em dia, uma solução comum para a realização de aplicações de automação industrial. Recentemente tem existido uma pressão do mercado no sentido de permitir às redes do tipo *fieldbus* de suportarem na mesma rede, nós sem fios (*wireless*, em inglês) e nós cablados, que tem sido reforçada pelos recentes desenvolvimentos tecnológicos na área da computação ubíqua e dos sistemas computacionais embarcados. Esta tese aborda os aspectos relacionados com a proposta de uma arquitectura inovadora, que permita a realização de um sistema de comunicação híbrido (*wired*/*wireless*) do tipo *fieldbus*, no qual possam operar nós cablados e nós sem fios.

O projecto Europeu RFieldbus, executado entre os anos de 2000 e 2002, foi uma importante iniciativa no sentido da concepção de um sistema de comunicações industriais do tipo *fieldbus* híbrido suportado pela tecnologia PROFIBUS (acrónimo de PROcess FIeld BUS). Os argumentos que foram utilizados para justificar a tecnologia utilizada continuam válidos na nossa opinião.

Num sistema RFieldbus, a ligação entre componentes cablados e componentes sem fios é feita através de dispositivos de interligação que operam ao nível da Camada Física (como repetidores). Nesta tese é proposta uma abordagem alternativa na qual os dispositivos de interligação operam como pontes (*bridges*, em inglês), e por isso ao nível da Camada de Ligação de Dados.

A tese que pretendemos defender é a de que tal abordagem é exequível e apresenta, face à abordagem RFieldbus, vantagens do ponto de vista do cumprimento de requisitos temporais e principalmente do ponto de vista da fiabilidade. Em consequência, esta tese propõe o *Inter-Domain Protocol* (IDP), que permite a execução de transacções entre estações pertencentes a meios diferentes. O IDP define o funcionamento das *bridges* quando processam este tipo de transacções, como é que a resposta pode ser obtida da estação destino (localizada noutro meio) e o formato das tramas trocadas entre as *bridges.* O IDP baseia-se nas características operacionais da Camada de Aplicação do PROFIBUS – o PROFIBUS-DP, desta forma garantindo a compatibilidade entre o sistema proposto e os dispositivos existentes.

As estações que comunicam utilizando tecnologia sem fios podem mover-se entre diferentes células. Consequentemente, esta tese propõe um mecanismo que permite a mobilidade de estações – o *Inter-Domain Mobility Procedure* (IDMP). Este procedimento é transparente para as aplicações do sistema dado que garante que durante o processo de mobilidade não existem erros e que não existe inversão na ordem das tramas. No entanto, o seu impacto no funcionamento do sistema resulta em atrasos adicionais para as transacções devido aos períodos de inacessibilidade da rede resultantes to IDMP. Assim, esta tese também propõe uma análise temporal que permite provar que a arquitectura proposta garante o funcionamento de aplicações com requisitos de tempo-real. Adicionalmente, são também mostradas as vantagens da arquitectura proposta em relação à arquitectura RFieldbus, nomeadamente em termos de recuperação de erros, contenção (dentro dos domínios) de falhas e melhor resposta temporal no caso de transacções entre estações do mesmo domínio,

Palavras chave: Sistemas de Tempo-Real; Comunicações de Tempo-Real; Redes Industriais.

Une Approche d'Anneaux Logiques Multiples pour les Réseaux PROFIBUS Temps-Réel à Capacité Sans-Fil

Résumé

Les systèmes de communication des réseaux de terrain (*fieldbus*) sont devenus une solution commune pour l'interconnexion des capteurs, actionneurs et des dispositifs de commande dans les applications d'automates industrielle et de contrôle de processus. Récemment, beaucoup d'efforts ont visé à étendre les fonctionnalités des réseaux de terrain pour supporter des infrastructures filaires et sans fil au sein du même réseau de communication. Cette thèse s'adresse aux problèmes liés à la proposition d'une nouvelle architecture de système de communication hybride dans les réseaux de terrains, permettant la cœxistence des médiums de transmission filaire et sans fil.

Le projet Européen RFieldbus présentait un effort important dans la proposition d'une solution de système de communication hybride pour les réseaux de terrains. Bien que certains résultats puissent être appliqués à d'autres standards de réseaux de terrain, la cible majeure des effort du projet était dévouée à une implémentation réelle dans les réseaux PROFIBUS. A notre avis, les arguments qui ont favorisé le déploiement de PROFIBUS comme le système de communication fédérant sont encore valides.

Dans RFieldbus, l'interconnexion des composants filaires et sans fil est basée sur l'interconnexion des dispositifs au niveau de la couche physique. Dans cette thèse, nous proposons une nouvelle approche où les dispositifs d'interconnexion agissent comme des ponts, et fonctionnent au niveau de la couche liaison de données.

L'hypothèse est qu'une telle approche est concevable et présente beaucoup d'avantages en terme de garantie temps-réel et de fiabilité par rapport a l'approche RFieldbus.

Pour cette raison, cette thèse propose le Protocole Inter-Domaines IDP (*Inter-Domain Protocol*) qui permet l'exécution des transactions entre les stations connectées sur différents types de médiums de transmission. Le protocole IDP spécifie le comportement des ponts lors du traitement des telles transactions, le mécanisme pour obtenir la réponse de la station répondeuse (*responder station*) se trouvant sur un autre médium de transmission, et le format des trames échangées entres les ponts. (I didn't understand the sentence just after)

Les stations mobiles sans fil sont capables de se déplacer d'une cellule à une autre. Par conséquent, cette thèse présente la Procédure de Mobilité Inter-Domaines IDMP (*Inter-Domain Mobility Procedure*). La procédure proposée est transparente aux applications du système, tout en garantissant que pendant sa progression, elle ne génère aucune erreur et qu'elle ne produit pas une inversion dans l'ordre des trames. Pour cette raison le protocole est compatible avec les stations standard de PROFIBUS. Son impact sur le réseau de communication se traduit par des délais supplémentaires sur les messages des transactions engendrés par des périodes d'inaccessibilité du réseau. Pour cela, une analyse des performances temporelles des protocoles IDP et IDMP est proposée dans cette thèse, montrant ainsi la capacité de l'architecture proposée de supporter des applications temps-réel.

Mots-clés: Systèmes Temps-Réel; Communications Temps-Réel; Réseaux Industriels, PROFIBUS.

Esta Tese é dedicada à
Anita,
à Inês e ao Tiago

Acknowldgements

First I would like to thank my supervisor, Eduado Tovar, for his essential reviewing work and his capability on helping me to overcome the obstacles during the development of this work, and also for his vision, leadership and hard work at the "helm" of the IPP-HURRAY! research group. I would also like to thank the support provided by my co-supervisor, Francisco Vasques.

Equally, I would like to thank the members of the doctoral committee for the time invested in the evaluation of this thesis.

During my thesis I have greatly benefited from the reviewing, contributions and fruitful discussions with Mário Alves. The work and the results provided by Paulo Sousa were also invaluable for the conclusion of this thesis. For both of them a very special thanks. Furthermore, I would like to stress that the discussions and moment of relaxation with my friends and colleagues at IPP-HURRAY! were very important.

To my colleagues at the Department of Informatics I would also like to thank their constant encouragement and attention. In particular to Berta Batista, Adriano Lhamas, Isabel Praça, Bertil Marques, Orlando Sousa, Alexandre Bragança and Maria João Viamonte.

I would also like to express my gratitude to all my friends for their encouragement and advices, especially to Armando Sousa and Claudia Ramalho for their contributions.

Thanks also to the Polytechnic Institute of Porto (IPP) and to its Scholl of Engineering (ISEP) for the institutional support provided. I would like to acknowledge the financial support provided by the Fundação para a Ciência e Tecnologia (FCT) through its PRODEP program.

I must acknowledge the support given by my parents, not only during the prosecution of this work but also throughout my entire academic career.

Finally, to my wife Anita, for her continuous support and understanding and also to my suns, Inês and Tiago, for the moments of happiness in family.

Porto, 10 November 2005

Luis Miguel Moreira Lino Ferreira

Table of Contents

Chapter 1 – Overview 1
1.1 Introduction 1
1.2 Research Context 2
1.3 Hypothesis and Research Objectives 5
1.4 Research Contributions 7
1.5 Structure of the Thesis 7

Chapter 2 – Technological Context 9
2.1. Introduction 9
2.2. Relevant Details on PROFIBUS 9
2.2.1. General Features 9
2.2.2. Data Link Layer (DLL) 10
2.2.3. Application Layer (AL): PROFIBUS-DP 15
2.3. Relevant Details on RFieldbus 19
2.3.1. Message Transactions and Basic Parameterization 21
2.3.2. Supporting Inter-cell Mobility 23
2.4. Some other Related Works and Technologies 25
2.5. Summary 26

Chapter 3 – Basics on PROFIBUS Timing Analysis 27
3.1. Introduction 27
3.2. Results Available for the Single Logical Ring Approach 28
3.3. Some Contributions 31
3.3.1. Comments on the Results Presented in (Cavalieri et al., 2002) 31
3.3.2. Latencies Related to the Entry of a Master into the Logical Ring 31
3.3.3. Accounting for the Mobility Mechanism on the Repeater-based Approach. 34
3.4. Summary 35

Chapter 4 – Aspects of the System Architecture, Network and Message Models 37
4.1. Introduction 37
4.2. Basic Aspects of the System Architecture 37
4.2.1. Types of End Systems 38
4.2.2. Types of Intermediate Systems 39
4.3. Basic Aspects of Bridge-based Intermediate Systems 42
4.4. Main Issues on the Protocol Extensions 44
4.4.1. Supporting Inter-Domain Transactions 44
4.4.2. Supporting Inter-Domain Mobility 46
4.5. Network Model 47
4.5.1. Model for the Communication Domains (D) 48
4.5.2. Model for the Master Stations (M) 50
4.5.3. Model for the Slave Stations (S) 51
4.5.4. Model for the Intermediate Systems (IS) 52
4.6. Message Stream Model (S_i^k) 53

4.7. Instantiating the Network and Message Models to an Example ... 54
4.8. Summary ... 57

Chapter 5 – The Inter-Domain Protocol (IDP) ... 59
5.1. Introduction ... 59
5.2. Bridge Architecture ... 60
5.2.1. Adapted MAC ... 61
5.2.2. Routing Tables (RT) ... 61
5.2.3. List of Open Transactions (LOT) ... 62
5.2.4. Frame Formats ... 63
5.2.5. Error Control ... 65
5.3. Cooping with Compatibility between IDP and PROFIBUS-DP ... 67
5.4. IDP Implementation Approach ... 70
5.4.1. `Indication_Handler` Function ... 70
5.4.2. `Confirmation_Handler` Function ... 73
5.5. Example Scenario ... 74
5.6. Summary ... 76

Chapter 6 – The Inter-Domain Mobility Procedure (IDMP) ... 79
6.1. Introduction ... 79
6.2. Description of the IDMP ... 80
6.2.1. Problems to be Addressed by the IDP ... 80
6.2.2. Agents of the IDMP ... 82
6.2.3. Phases of the IDMP ... 83
6.2.4. Enabling the Mobility of Mobile Wired Domains ... 87
6.2.5. Routing Tables Operation with IDMP ... 88
6.3. Example Scenario ... 88
6.4. Details on the IDMP Agents Implementation ... 92
6.4.1. State Machine for the Global Mobility Manager ... 92
6.4.2. State Machine for the Domain Mobility Manager ... 92
6.4.3. DMM MAC State Machine ... 94
6.4.4. Mobility-related Bridge Master Functionalities ... 95
6.5. IDMP Messages ... 96
6.6. Summary ... 97

Chapter 7 – Timing Analysis of the IDP and the IDMP ... 97
7.1. Introduction ... 97
7.2. WCRT analysis for IDT transactions ... 97
7.2.1. WCRT Analysis of IDTs Based on SDA or SDR Services ... 97
7.2.2. WCRT Analysis of IDTs Based on the SDN Service ... 101
7.2.3. Working on Reducing the Pessimism ... 102
7.3. IDMP Timings ... 105
7.3.1. Phase 1 ... 106
7.3.2. Phase 2 ... 107
7.3.3. Phase 3 ... 109
7.3.4. Phase 4 ... 111
7.3.5. Worst-Case IDMP duration ... 112

7.3.6. Computing the Number of `Beacons` 112
7.4. Summary 113

Chapter 8 – WCRT Analysis of Transactions Considering the Latencies of the IDMP 115
8.1. Introduction 115
8.2. Inaccessibility Periods due to the IDMP 115
8.2.1. Time During which IADT are Disabled in a Domain 116
8.2.2. Time During which IDT are Disabled in a BM 117
8.2.3. Time During which IDT Addressed to Mobile Wireless Stations are Disabled 118
8.2.4. Time During which Mobile Wireless Stations are Inaccessible 119
8.3. Incorporating the IDMP into the WCRT of Transactions 120
8.3.1. Intra-Domain Transactions (IADT) 120
8.3.2. Inter-Domain Transactions (involving Domain Resident Wireless Stations or Wired Stations) 121
8.3.3. Transactions Involving Mobile Wireless Stations 125
8.4. Summary 128

Chapter 9 – Numerical Examples and Performance Comparisons 129
9.1. Introduction 129
9.2. Network Example 130
9.3. Computing the Duration of Message Transactions 134
9.3.1. Obtaining the Number of Message Streams Relayed by the BMs 135
9.3.2. Computing the WCRT for IDTs 136
9.4. Mobility-related Timings 137
9.4.1. Phase 1 137
9.4.2. Phase 2 137
9.4.3. Phase 3 138
9.4.4. Phase 4 139
9.5. IDMP Impact on the WCRT of Transactions 141
9.5.1. Inaccessibility Periods 141
9.5.2. WCRT for the System Message Streams 143
9.6. Results Extracted from Simulation 143
9.6.1. Basics on the Simulator Architecture 144
9.6.2. Simulation Parameters and Operational Characteristics Assumptions 145
9.6.3. Simulation Results 146
9.7. Comparing with the Repeater-based Approach 150
9.7.1. Equivalent Repeater-based Network Topology 150
9.7.2. Variability of the WCRT 151
9.7.3. Comparing the Responsiveness to Errors in Both Approaches 155
9.8. Summary 158

Chapter 10 – Conclusions and Future Work 159
10.1. Research Context and Objectives 159
10.2. Main Research Contributions 160
10.2.1. Hybrid Wired/Wireless PROFIBUS Bridge-based Architecture 160

10.2.2. Inter-Domain Protocol (IDP) 160
10.2.3. Inter-Domain Mobility Procedure (IDMP) 161
10.2.4. Timing Analysis 161
10.2.5. Comparing with the RFieldbus Approach 162
10.3. Future work 162

References 165

Annex – Acronyms and Symbols 169
A.1. Acronyms 169
A.2. Symbols 171

Chapter 1

Overview

Fieldbus communication systems have become a common solution to the problem of interconnecting sensors, actuators and control devices in manufacturing automation and process control applications. Recently, there has been an enormous eagerness to extend fieldbus functionalities to support wired and wireless network stations in the same network system. This thesis addresses the issues related to the proposal of a novel architecture for a hybrid fieldbus communication system, where wired and wireless transmission media coexist. This chapter presents the context, defines the hypothesis, summarises the main contributions and provides a view on the overall organisation of the thesis.

1.1 Introduction

The constant evolution of Information and Communication Technologies (ICT) has driven their use in a widespread range of applications, from large information systems to small and powerful devices. Personal Digital Assistants (PDA) and automatic vending machines are just two examples of mobile computing devices used in our everyday life. Similarly, also industrial automation systems have been benefiting substantially from ICT.

Nowadays, industrial automation systems usually follow distributed computer-controlled approaches. This setting is often referred to as a Distributed Computer-Controlled System (DCCS). In such systems, different hardware and software modules co-operate in order to achieve a common goal. Typically, this co-operation is supported by a communication infrastructure specially suited to fulfil specific requirements of industrial automation systems. This type of networks is known by the buzzword “fieldbus”.

Fieldbus networks are typically capable of fulfilling a set of requirements inherent to factory-floor environments. These include timeliness, reliability, cost-effectiveness and suitability of application protocols and services, just to mention some of the most relevant. Timeliness deserves further reasoning, since this requirement is omnipresent throughout this thesis. A real-time (distributed) computing system is a system in which correctness depends not only on the logical results of computation, but also on the time at which results are produced (Stankovic, 1988). Therefore, emerging and future fieldbus networks must be able to provide the means for guaranteeing the timeliness requirements imposed by the distributed applications.

Additionally, the factory-floor is becoming more and more sophisticated, with increasingly demanding computing devices proliferating everywhere (Pacheco and Tovar, 2002; Tovar *et al.*, 2003). A few examples are automatic transportation systems such as Automatic Guided Vehicles (AGVs), or handheld devices such as PDAs used for process monitoring, or even wearable computers used to provide plant-floor operators with augmented reality capabilities.

In such ubiquitous computing factory-floor, a new challenge emerges to fieldbus networks: the ability to provide seamless operation between wired and wireless stations, while still being capable of fulfilling important application requirements such as timeliness or reliability.

1.2 Research Context

The RFieldbus European project (Alves *et al.*, 2002; Rauchhaupt, 2003) was one major effort towards a hybrid wired/wireless fieldbus solution. Although some of the achievements could potentially be applied to other commercial-off-the-shelf (COTS) fieldbus networks, most of the effort in that project was devoted to an implementation over PROFIBUS (acronym for PROcess FIeld BUS) technologies (EN50170, 1996).

The arguments that were put forward in favour of using PROFIBUS as the federating communication system are still valid, in our opinion. In fact, PROFIBUS is the world's leading fieldbus standard for manufacturing automation and process control, with over 20% of the market share, and several millions of devices in operation worldwide. Additionally, PROFIBUS offers one of the fastest transmission speeds available in a fieldbus system: 12 Mbps. Although transmission speed is not synonymous of real-time ability, it still is an important characteristic for distributed applications imposing stringent real-time requirements. PROFIBUS has another interesting feature. It is designed to provide different qualities of service in terms of timeliness, providing intrinsic mechanisms that distinguish the way high and low priority messages are transmitted. Moreover, the PROFIBUS Medium Access Control (MAC) protocol, being based on the measurement of the real token rotation time, induces a well-defined timing behaviour to the worst-case message response time, since the upper bound for the actual token rotation time can be know *a priori* (Tovar and Vasques, 1999b). Therefore, the PROFIBUS protocol is able to support guaranteed real-time traffic.

Another set of requirements defined for the RFieldbus approach included the need to provide compatibility with legacy wired PROFIBUS technologies, while at the same time avoiding the need for complex mechanisms for enabling the interoperability between wired and wireless stations. Therefore, the architecture of the RFieldbus system was based on the option that all network stations listen to every transmitted message – a "broadcast" network, and belong to the same logical ring, i.e. a single token rotating among the masters in the network, regardless of their wired or wireless nature. This approach is denoted as a Single Logical Ring (SLR) approach, in this thesis.

To give a better intuition of the RFieldbus approach, an example system is presented in Figure 1.1. The example system is composed of 3 master stations: M1, M2 and M3. M2 is a wireless master station albeit not mobile, while M3 is a mobile wireless master station which can move in the range of radio cell 1 and radio cell 2. The overall system contains 7 slave stations, denoted as S1 – S7. From these, S4 and S7 are wireless slave stations, with the latter being mobile. In RFieldbus, the interoperability between wired and wireless stations relies on the use of Intermediate Systems (IS), which are interconnecting devices operating essentially as repeaters, and therefore operating at the

Physical Layer (PhL) level. In the outlined example, 3 of these repeater devices are considered: REP1, REP2 and REP3. Among other functionalities, these repeaters perform reciprocal frame and bit rate conversion (at the PhL level) between wired and wireless media.

In the example, all communications between wireless stations operating in a specific radio cell (including the wireless interface of the repeaters) are relayed through Base Station (BS) devices. Each BS structures a radio cell, and overlapping radio cells must operate in different radio channels, each radio comprising one uplink (to the BS) and on downlink channel (from the BS).

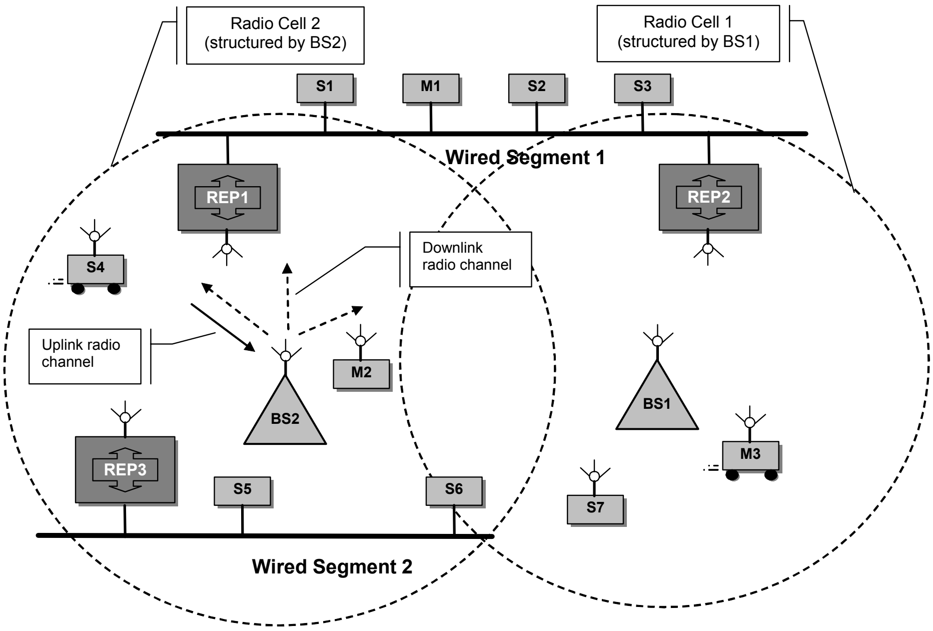


Figure 1.1 – Example of a hybrid wired/wireless RFieldbus network

In this setting, and as already mentioned, all messages transmitted either by the masters (e.g., token or message requests to slaves) or by the slaves (e.g., responses to masters' requests) are "broadcast" throughout the overall network. Moreover, all masters in the network belong to the same logical ring. For this particular example, the token rotation can have the following sequence: … → M1 → M2 → M3 → M1… .

In RFieldbus, inter-cell mobility is supported, and is implemented in a very simple and efficient way. Periodically, one specific master in the system (denoted as Mobility Master) emits a special non-acknowledged request: the `Beacon Trigger`. This message is received by all base stations in the system, which in turn start to transmit `Beacons` in their respective radio channels. When the wireless stations receive the `Beacon Trigger`, they start assessing the quality of the different radio channels operating in the network. At the end of this assessment phase, wireless stations switch to the channel with the best quality. During this “handoff” phase, there are no PROFIBUS frames circulating in the network, since the mobility master holds the token until the channel assessment is finalised (in all wireless stations in the system). For this procedure to work properly, and in order not to jeopardise normal PROFIBUS traffic, an upper bound for the “handoff” time span must be calculated *a priori*, and used as a time parameter which is set in the mobility master.

Due to the broadcast nature of the network, other timing parameters must also be properly set for the system to work correctly. Firstly, in order to cope with different bit rates and frame formats (wired and wireless frame formats) at the PhL level, there is the need to increase the value of the `Idle Time` PROFIBUS parameters between consecutive message transactions. In PROFIBUS, a message transaction usually corresponds to a request issued by a master and the related response issued by a slave. This kind of traffic adaptation scheme avoids increased queuing delays in the repeaters, and also enables the computation of an upper limit for message turnaround times. Nevertheless, communication latencies will increase with the number of repeaters in the path between the initiator and the responder of the transaction.

As a result, in RFieldbus there is the need to set the masters’ parameter (the `Slot Time`) with a value large enough to encompass the larger time span between the end of transmission of message requests and the start of reception of message responses. The side effect is that responsiveness to system errors (either token loss or transmission error) becomes smaller than for a single segment network.

Another aspect which may be seen as a drawback in the RFieldbus approach concerns fault containment. An error such as a token loss will have repercussions in the overall network. The reader is referred to (Alves *et al.*, 2002; Alves, 2003) for further details on the mechanisms and characteristics of the RFieldbus approach, which, in any case, will be addressed in more detail in Chapter 2 of this thesis.

In summary, while at one hand the RFieldbus approach offers simplicity and compatibility as major advantages, it also has a few drawbacks, which become more acute as the number of masters and the number of different segments (and repeaters) increase, or as heterogeneity (bit rate, frame formats) between wired and wireless media increases. These drawbacks can be summarised as follows:

- no fault containment (for example, a token loss implies malfunction in the whole system);
- larger values for the master’s `Slot Time` parameter implies lower responsiveness of the network to token error or even to transmission errors;
- extra inserted `Idle Time`, in the masters, leads to longer response times for message transactions, and therefore lower ability to cope with more stringent real-time applications, even if the message transactions are between stations in the same segment.

1.3 Hypothesis and Research Objectives

An intuitive alternative for the Single Logical Ring (SLR) hybrid wired/wireless PROFIBUS network described in Section 1.2 would be a Multiple Logical Ring (MLR) approach, where bridges would be used as Intermediate Systems (IS), instead of repeaters

Bridges are intermediate systems that operate at the Data Link Layer (DLL) level. Assuming a two-port bridge interconnecting two different network segments, frames arriving to one bridge port are only relayed to the other port if the destination address embedded in the frame corresponds to a MAC address of a station physically reachable through that other port.

With a MAC protocol as the one used in PROFIBUS (timed token passing), a bridge needs to have two network interfaces, both supporting the same DLL and specifically the same MAC protocols. Nevertheless, physical layers could be distinct, as for the case of hybrid wired/wireless networks. This means that such a dual-port PROFIBUS bridge would contain two master stations, and the network example illustrated in Figure 1.1 would now look like as illustrated in Figure 1.2.

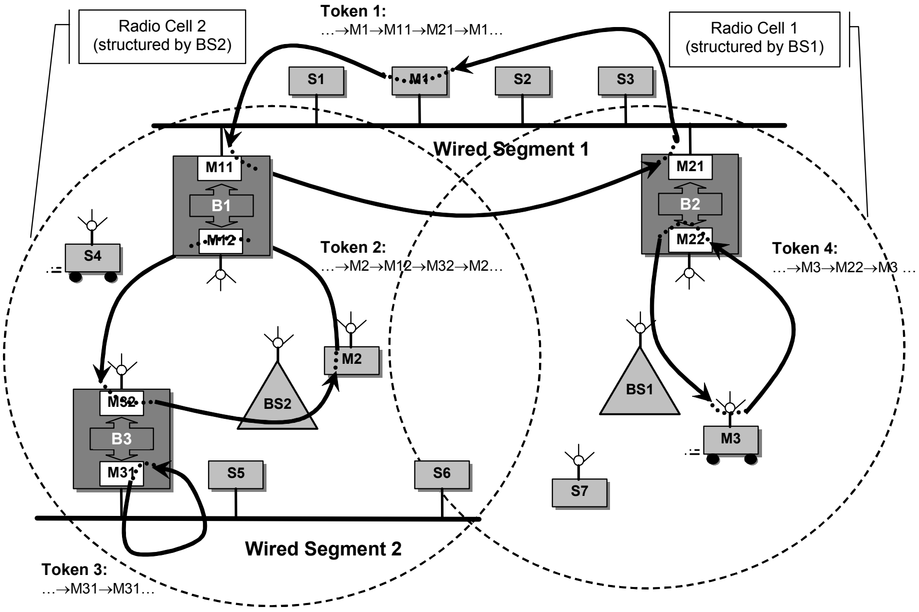


Figure 1. 2 – Example of a hybrid wired/wireless PROFIBUS network using Bridges as intermediate systems

In the exemplified Multiple Logical Ring (MLR) PROFIBUS network, three bridges are used: B1, B2 and B3. Each bridge has two master PROFIBUS interfaces. The pairs M11 and M12, M21 and M22, and M31 and M32, are associated to B1, B2 and B3, respectively.

The three bridges interconnect 4 logically separated PROFIBUS networks, thus leading to 4 independent tokens and correspondent token rotation schemes. These token rotation schemes could be as follows:

– Token 1: …M1 → M11 → M21 → M1 …;

- Token 2: …M2 → M12 → M32 → M2 …;
- Token 3: …M31 → M31 …;
- Token 4: …M3 → M22 → M3 ….

However, if M3 moves from radio cell 1 to radio cell 2, the token rotation schemes (illustrated in Figure 1.3) would become:

- Token 1: …M1 → M11 → M21 → M1 …;
- Token 2: …M2 → M3 → M12 → M32 → M2 …;
- Token 3: …M31 → M31 …;
- Token 4: …M22 → M22 ….

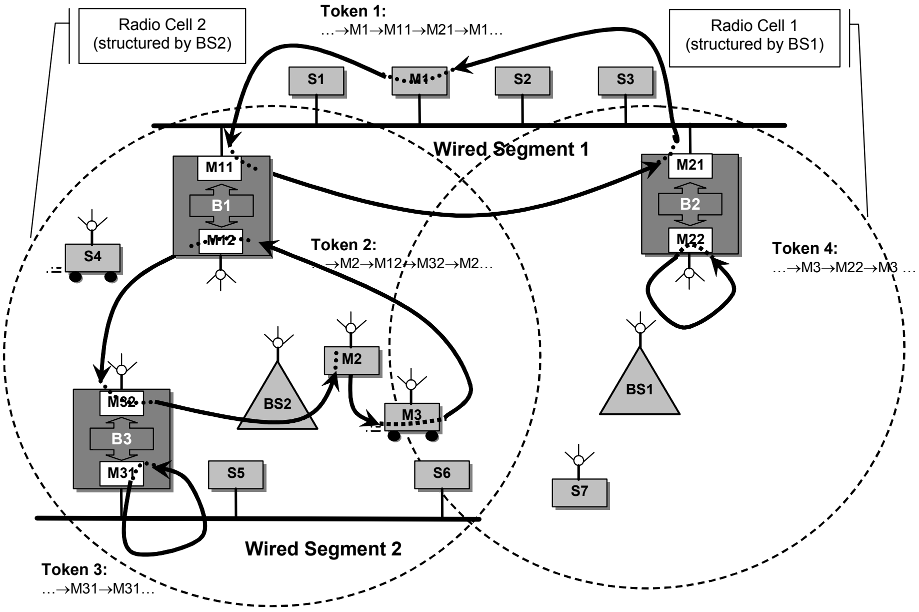


Figure 1.3 – Example of a hybrid wired/wireless PROFIBUS network using Bridges as intermediate systems (M3 moved to radio cell 2)

Theoretically speaking, the advantages of such a bridge-based approach, when compared to the RFieldbus approach, are potentially the following:

- there is an important level of fault isolation between network segments (formed by each logical ring);
- `Slot Time` parameters need not to be set taking into account global network latencies but only single segment latencies instead, meaning more responsiveness to errors;
- there is no need to insert extra `Idle Time` between two consecutive message transactions to perform media adaptation, as in RFieldbus, and therefore message transactions between network stations in the same network segment have smaller worst-case response times.

The hypothesis is that such a MLR approach is devisable, while guaranteeing total compatibility with the existing PROFIBUS standard and coping with the original real-time capabilities of PROFIBUS. Moreover, we aim at demonstrating that such an approach will feature the previously listed advantages over RFieldbus, the current state-of-the-art solution.

To tackle this challenge, a number of research objectives must be addressed. Firstly, and since the PROFIBUS standard does not define any bridging mechanisms, these need to be specified. This specification must include the architecture of the bridging devices, a protocol to handle message transactions between stations pertaining to different logical rings, as well as the required mechanisms to support transparent mobility of stations between radio cells. Moreover, a proper timing analysis must be devised on the proposed protocols and mechanisms, with the purpose of enabling guaranteed (prior to run time) worst-case real-time behaviour of the proposed system architecture.

1.4 Research Contributions

The main research contributions of this thesis are listed bellow.

1. The definition of a hybrid wired/wireless PROFIBUS-based architecture where the interconnection between different media is achieved by intermediate systems operating as bridges (Ferreira *et al.*, 2002).
2. The specification of a protocol which transparently enables the execution of transactions between stations in different domains – the Inter-Domain Protocol (IDP) and the mechanisms which support the operation of the bridges (Ferreira *et al.*, 2003b).
3. The definition and specification of mechanisms to transparently support the mobility of stations between different wireless cells – the Inter-Domain Mobility Procedure (IDMP) (Ferreira *et al.*, 2003a).
4. A worst-case message response time (WCRT) analysis for the overall system, in a way that real-time communications can be guaranteed for such a hybrid networking system. This timing analysis takes into consideration the mobility procedure, since it impacts on worst-case response times (Ferreira and Tovar, 2004a; Ferreira and Tovar, 2004b).

1.5 Structure of the Thesis

The structure of this thesis is as follows.

In Chapter 2, we survey the relevant technological context, with special emphasis given to PROFIBUS and RFieldbus technologies. In Chapter 3, the most relevant previous work on temporal analysis of PROFIBUS networks is surveyed. We discuss some of the formulations and, in some cases, propose some improvements. These works are then extended in order to encompass the dynamic management of logical rings (e.g., a master joining a logical ring). The analysis provided in this chapter is later used in

Chapters 7 and 8 as a basis for devising a worst-case timing analysis for the proposed system architecture.

Chapter 4 provides the general characterisation of the proposed system architecture, as well as the definition of the network and message models. It starts by introducing the components of the proposed hybrid wired/wireless network along with some rationale for supporting the bridge-based approach. Then, it presents an overview of the major issues on the proposed protocol, which are later addressed in detail in Chapters 5 to 8. Finally, analytical models for the different network components are proposed, which are relevant for better understanding the timing analysis performed in Chapters 7 and 8.

Chapter 5 focuses on the description of the Inter-Domain Protocol (IDP). It starts by detailing the architecture of the bridging devices and the formats of the Inter-Domain Frames (IDF) that are embedded in standard PROFIBUS frames. To illustrate the operation of the IDP, an example scenario is presented. The chapter also includes some important aspects related to the implementation of the protocol.

Chapter 6 describes the proposed extensions to the IDP protocol for enabling inter-domain mobility, by providing a detailed description and reasoning of the Inter-Domain Mobility Procedure (IDMP). The IDMP is a mechanism which is driven by two major agents – the Global Mobility Manager (GMM) and the Domain Mobility Manager (DMM). An example scenario and an implementation approach are also provided.

The support of distributed real-time applications requires that communication delays are known and bounded. Chapter 7 provides a detailed timing analysis of the IDP protocol, based on the analysis presented in Chapter 3. The analysis presented in Chapter 7 does not take into account the latencies and network inaccessibility periods caused by the IDMP. Consequently, Chapter 8 extends these results by considering the impact of the IDMP, therefore providing analytical tools enabling engineering bridge-based systems where stations are allowed to move between radio cells.

Chapter 9 exercises and explores a set of numerical examples which illustrate how the timing analysis developed in Chapters 7 and 8 can be applied to hypothetical networking system scenarios. It also presents some results which were obtained by simulation and compares them with the results from the analytical formulations. Additionally, this chapter discusses the main sources of pessimism related to the proposed timing analysis and how that pessimism can be reduced. With these results, we demonstrate the advantages (and disadvantages) of our approach against the RFieldbus approach.

Finally, Chapter 10 summarises the contributions of this thesis, provides conclusions, and describes some lines of work that can potentially be explored as a natural sequence of the work described in this dissertation.

Chapter 2

Technological Context

This chapter provides an overview of some relevant communication technologies related to the framework of this thesis. Since the PROFIBUS protocol is used as the federating communication system for the proposed hybrid architecture, its main characteristics are addressed with some detail. Special relevance is also given to the RFieldbus approach, to which our proposal should be compared.

2.1. Introduction

As stressed in Chapter 1, there has been an enormous eagerness for supporting wireless and mobile communications in fieldbus networks. While completely new fieldbus architectures could be devised from scratch with these requirements in mind, most of the research efforts (including ours) focus on specifying architectures based on already existent and widespread COTS (Commercial Off-The-Shelf) technologies.

At the light of the results of the RFieldbus project, namely from the experience resulting from the two field trials (Tovar *et al.*, 2003), PROFIBUS and IEEE 802.11b (IEEE 802.11b, 1999) proved to be a good choice for structuring a hybrid wired/wireless fieldbus communication system, which are considered still valid.

In this chapter, we describe the most relevant characteristics of PROFIBUS (Section 2.2) and RFieldbus (Section 2.3). The objective is to provide the reader with the necessary background and intuition for tackling the remaining chapters of this thesis. Even not being crucial for the research objectives and technological framework of this thesis, Section 2.4 briefly surveys some other recent and ongoing research efforts related to the use of wireless technologies in the factory-floor.

2.2. Relevant Details on PROFIBUS

2.2.1. General Features

PROFIBUS was standardised in 1996 as an European standard (General Purpose Fieldbus Communication System - EN50170). It is based on the International Standards Organisation (ISO) Open System Interconnection (OSI) reference model, however collapsed to just three layers: Physical Layer (PhL), Data Link Layer (DLL) and Application Layer (AL). There is also a transversal management functionality called

Fieldbus Management (FMA1/2), which is responsible for the management of the layers 1 and 2, the PhL and the DLL, respectivly.

The PROFIBUS PhL can use the RS-485 standard over twisted pair or coaxial cable for the transfer of data, with bit rates up to 12 Mbit/s. For special applications, it is also possible to use other types of physical media, like optical fibre, power cable or RS-485-IS (for intrinsically safe applications).

The PROFIBUS DLL uses a token passing procedure (Grow, 1982) to grant bus access to masters, and a master-slave procedure used by masters to communicate with slaves (or other masters). Slaves do not have communication initiative. They are only capable of transmitting a response (or an acknowledgement) upon master request. The token is passed between masters in ascending Medium Access Control (MAC) address order, thus the masters organise network access in a logical ring fashion.

The PROFIBUS standard considers two different types of Application Layer profiles: PROFIBUS-FMS (Fieldbus Message Specification), which is being abandoned due to design complexity and cost, and PROFIBUS-DP (Decentralised Peripherals), which is being increasingly adopted for industrial automation and process control applications. PROFIBUS-DP is particularly suited for the cyclic exchange of data between master (Programmable Controllers, PC, etc.) and slave devices (valves, I/O devices, drives, etc.).

2.2.2. Data Link Layer (DLL)

Message Cycle

In PROFIBUS, only master stations may initiate transactions, whereas slave stations do not transmit on their own initiative, but only upon (master) requests. The station that sends an *Action Frame* (the first frame transmitted in each transaction) is the *initiator* of the transaction, while the addressed one is the *responder*. A transaction (or message cycle) consists on the request or a send/request frame from the initiator (always a master station) and the associated acknowledgement or response frame from the responder (either a master station or a slave station, but typically a slave station).

All stations (except the initiator) monitor all the requests but will only acknowledge or respond if, and only if, they are the addressees in the initiator's request. Moreover, the acknowledgement or response frame must arrive before the expiration of the `Slot Time` (T_{SL}) a master DLL parameter, otherwise the initiator repeats the request a number of times defined by the `max_retry_limit`, another master's DLL parameter. If the station does not acknowledge or respond after that number of retries, the initiator marks that station has having problems. After, when this initiator makes other requests to the same station, it does not make any retries until the station responds or acknowledges again.

Token Passing

The token is passed between masters in ascending address order. The only exception is that in order to close the logical ring, the master with the highest address must pass the token to the master with the lowest one. Each master knows the address of the previous station (PS – `Previous Station` address), the address of the following station (NS

– `Next Station` address) and, obviously, its own address (TS – `This Station` address).

If a master station receives a token addressed to itself from a station registered in the `List of Active Stations` (LAS) as its predecessor (PS = TS) then this master becomes the token owner, and may start processing message cycles. On the other hand, if a master receives the token from a station which is not its previous station, it assumes that an error has occurred, and it will not accept the token. However, if it receives a subsequent token from the same station, it accepts the token and assumes that the logical ring has changed. In this case, it updates the original PS value by the new one in its LAS table.

If after transmitting the token frame and after the expiration of the `Synchronous Time` (idle bus for a 33 bits period) within the `Slot Time`, the master receives either a valid frame or an invalid one, it assumes that its successor owns the token. Therefore, it ceases monitoring the activity on the bus. In case 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 finds a successor from its `List of Active Stations`.

Token Cycle

After receiving the token, a master station is allowed to execute message cycles for a duration of `Token_Holding` time (T_{TH}), which is equal to the difference, if positive, between the `Target_Token_Rotation` time (T_{TR}) and the `Real_Rotation` time (T_{RR}). T_{TR} is a parameter common to all masters in the network, which must be set to the expected time for the token cycle. T_{RR} is the time measured between two consecutive token receptions – the token cycle.

PROFIBUS defines two main categories of messages: high-priority and low-priority, each using a different transmission queue that is handled differently by the DLL. At the arrival of the token, the T_{TH} timer is loaded with the value corresponding to the difference between T_{TR} and T_{RR}. If the token is delayed, then T_{TH} is set to zero and the master is only allowed to perform, at most, one high-priority message transaction. Otherwise, the master is allowed to perform high-priority message transactions until the value of the T_{TH} timer becomes negative. Low-priority messages are only transmitted when the high-priority queues are empty and T_{TH} is still positive. Note that once a message cycle is started it is always completed, including any retries, even if in meanwhile T_{TH} expires.

Ring Maintenance

In order to maintain the logical ring, PROFIBUS provides a decentralised (in every master station) ring maintenance mechanism. Each PROFIBUS master maintains two tables: the `Gap List` (GAPL) and the `List of Active Stations` (LAS). It may also optionally maintain a `Live List` (LL) table.

The GAPL consists on the address range from address TS until NS. 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. Each master station in the logical ring starts to check its Gap addresses every time its `Gap Update` timer (T_{GUD}) expires. If a station acknowledges positively to the GAP request (a `FDL_Request_Status` frame), with the state `Not_Ready_to_Enter_ Logical_Ring` or `slave_station`, it is accordingly marked in the GAPL and the next address is checked. If a station answers with the state `Ready_to_Enter_ Logical_Ring`, the token holder changes its GAPL and passes the token to the new NS. This (master) station, which has newly been admitted to the logical ring, has already built up its LAS when it was in the `Listen_Token` state, so it is able to determine its GAPL and its NS. 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 the execution of all high-priority transactions, and if the value of the token holding timer, initially loaded with T_{TH}, is still positive.

The LAS table 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 reception of token frames.

Concerning the LL table, there is the need for an explicit demand from the DLL user, 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 LL table as existent master or slave stations.

Additionally, in order to enhance the communication system's reliability, PROFIBUS handles operational or error states, concerning logical ring management. Some of the more relevant are described next.

- Multiple tokens (in one segment). This situation may occur in case a master has a malfunctioning transceiver (e.g. a deaf receiver). While the master is in the `Listen_Token` state, it monitors the bus activity and, depending on whether it detects activity, it can claim the token or wait to be the addressee of a token frame. If the master has a deaf receiver, it assumes that there is no active master and elects itself as the active one. However, the DLL controller of the malfunctioning station monitors its own activity during the transmission of the token frame. Therefore, if it does not detect any activity due to the transmission of the token, it enters the `Offline` state, notifying the PROFIBUS DLL management entity – FMA 1/2.
- Lost token. This abnormal situation is clearly recovered by the DLL controller by means of a continuous monitoring activity performed by each master in the logical ring. If a period of inactivity longer then the Time-Out time (T_{TO}) is detected, then the token is claimed by the master with minor address previously (before the token loss) in the logical ring, and the logical ring is reinitialised. The master which claimed the token passes the token to itself and uses the Gap Update mechanism to include the other masters on the newly formed logical ring.

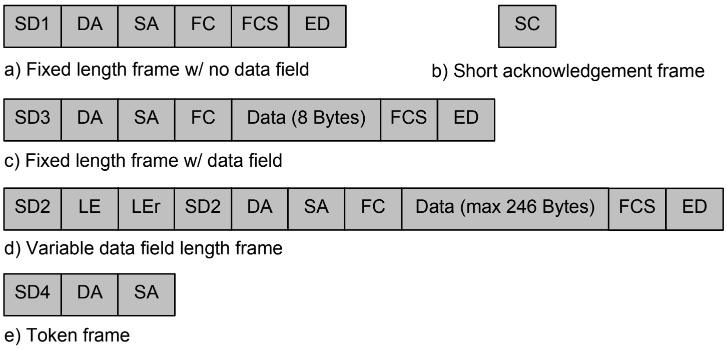
– <u>Error in token passing</u>. The DLL controller also provides mechanisms to recover from this situation. While the station's transceiver transmits the token frame, the DLL controller monitors the activity on the bus and if it does not detect any activity, it should enter the `Offline` state. The subsequent loss of token is recovered by the claiming procedure described above. The token passing procedure has a high level of reliability by itself. If the designated station does not respond, the master tries to pass the token to the next but one station in its LAS. On the other hand, if a station is taken from the ring not by its own initiative (i.e., in spite of being `Active_idle` in the logical ring, it does not receive any token frame), it will notify the event to FMA1/2.

The DLL controller also provides the specific services to inform the FMA1/2 about the occurrence of a malfunctioning in its transceiver and of multiple assignment of station addresses.

DLL Frame Formats

PROFIBUS DLL defines 3 types of request/response frames which are the `Fixed Length with no Data Field`, the `Fixed Length with Data Field` and the `Variable Data Field Length`, as illustrated in Figure 2.1.a), c) and d), respectively.

Each of these three types includes the following fields: `Destination Address` (DA), `Source Address` (SA), `Frame Control` (FC) and `Start Delimiter` (SDx). These frames also include the `Frame Check Sequence` (FCS) and the `End Delimiter` (ED).



a) Fixed length frame w/ no data field

b) Short acknowledgement frame

c) Fixed length frame w/ data field

d) Variable data field length frame

e) Token frame

Figure 2.1 – PROFIBUS DLL frame formats

Variable data field length frames additionally contain two `Data Length` fields (LE and LEr) and they can optionally include the `Destination Address Extension` (DAE) and `Source Address Extension` (SAE), in the `Data` field. These extension fields can be used to identify AL services which originated the frame, as well as destination services.

PROFIBUS also defines the Short aCknowledgement frame (SC) and the Token Frame, illustrated in Figure 2.1.b) and e), respectively. The first consists of a single byte frame, and it is used as negative or positive acknowledgement to a request.

Data Link Layer Services

PROFIBUS defines 4 types of data transfer services: Send Data with Acknowledge (SDA); Send Data with No acknowledge (SDN); Send and Request Data (SRD) and Cyclic Send and Request Data (CSRD).

The SDA service allows a user to transmit data to another station and receive a Short Acknowledge confirming its reception by the responder station. The SDN service permits to transfer data to a single station, to a group of stations (multicast) or to all stations (broadcast). The SRD service allows the transmission of a message to another station and the retrieval of a response. This service can be used, for example, to send the output settings for an I/O device and retrieve the state of the device's input ports. The CSRD builds upon the SRD service adding the capability of transferring data periodically, according to the user requirements. The CSRD service is usually not implemented in current commercial hardware platforms.

Timing Parameters

The PROFIBUS standard defines several timing parameters, some of which are relevant in the context of this thesis, such as the Idle Time and the Slot Time parameters, which are briefly explained next.

There are two Idle Time (T_{ID}) parameters - T_{ID1} and T_{ID2}. T_{ID1} is a period of inactivity, inserted by a master station, after an acknowledgment, response or token frame. This parameter must be set as follows:

$$T_{ID1} = \max\{T_{SYN} + T_{SM}, \min T_{SDR}, T_{SDI}\} \tag{2.1}$$

T_{SYN} (Synchronisation Time) is the minimum time interval for an idle bus state before a station may accept the beginning of an action frame or token. T_{SM} (safety margin) is the time that elapses after the end of the T_{SYN} which is required by the receiver circuitry to be ready to start receiving a frame. $minT_{SDR}$ is the minimum station delay of a responder. T_{SDI} is the station delay of the initiator, after which the initiator is ready to start receiving a frame from the responder. Figure 2.2 depicts an example where the Transmission Delay time (T_{TD}) due to the network propagation delay is also illustrated.

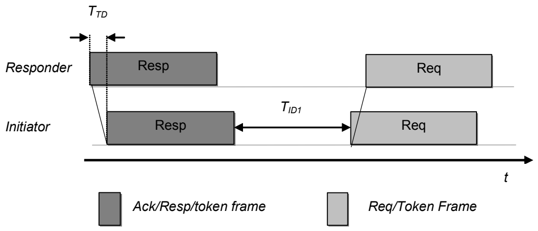


Figure 2.2 – Idle Time parameter – T_{ID1}

T_{ID2} is the idle time inserted by a master station after transmitting an unacknowledged request frame. T_{ID2} must be set as follows:

$$T_{ID2} = \max\{T_{SYN} + T_{SM}, \max T_{SDR}\} \tag{2.2}$$

where $maxT_{SDR}$ is the maximum delay of a responder station.

The `Slot Time` (T_{SL}) timer is used by a master station to detect if the communication with a slave (or with its successor, in the token passing) has failed. The T_{SL} timer is loaded with T_{SL} at the end of the transmission of a request frame. Upon its expiration, the master station may execute another retry for the same request, if the value of the number of retries executed is smaller than the `max_retry_limit` parameter, or it may inform the upper layers of a transmission failure. The timer is also loaded with T_{SL} after transmitting the token. If it expires before the master has detected any activity in the bus then it signals the MAC layer in order to take the appropriate actions.

The `Slot Time` parameter (T_{SL}) must be set to the maximum between two values – T_{SL1} and T_{SL2}. T_{SL1} can be calculated as follows:

$$T_{SL1} = 2 \times T_{TD} + \max T_{SDR} + 11bit + T_{SM} \tag{2.3}$$

where *bit* is the time duration of a bit. T_{SL2} can be calculated as follows:

$$T_{SL1} = 2 \times T_{TD} + \max T_{ID1} + 11bit + T_{SM} \tag{2.4}$$

Note that all masters in the network must hold the same T_{SL} value, due to the token passing mechanism.

In RFieldbus, the setting of the `Slot Time` and `Idle Time` parameters must be made differently, in order to encompass the latencies of the repeaters. Section 2.3.1 will address the methodology followed in RFieldbus.

2.2.3. Application Layer (AL): PROFIBUS-DP

The PROFIBUS-DP (DP for short) protocol is specially suited for the exchange of data between controllers (typically masters) and field devices like I/O, drives or valves (typically slaves). DP provides the functionalities to configure field devices and to perform cyclic exchange of data between the controller and the field devices.

DP is available in three versions (V0, V1 and V2), which are specified in IEC 61158 – Fieldbus Standard for Industrial Systems, PROFIBUS (type 3).

DP-V0 contains the main structural elements of PROFIBUS AL by providing the basic functionalities, including cyclic data exchange, station diagnosis, module diagnosis and channel-specific diagnosis. DP-V1 extends DP-V0 by adding services for acyclic data exchange, visualisation and alarm handling. DP-V2 adds features geared towards drive technology, like isochrounous slave mode and slave to slave communication.

As DP-V0 is the most widespread PROFIBUS technology and is the basis for the other versions, the remaining description is only devoted to that version.

The DP application layer operates with 3 types of stations: class 1 master, class 2 master and slaves. Class 1 masters are capable of controlling several slave devices and of communicating with them using a polling method. Class 2 masters are management devices that implement a set of functions to configure, manage and diagnose any other types of DP devices. PROFIBUS-DP only allows a slave to interoperate with a single master, while a master can interoperate with several slaves.

The main functionalities of PROFIBUS-DP are related to the reading and writing of variables from/to slave devices. The communication between a master class 1 and a slave starts by the parameterisation and configuration of the slave, after which is possible to retrieve data from the slave. The retrieval of data is made cyclically by the DP protocol, according to timing parameters configured by the user.

The data exchange services are of three types: configuration and parameterisation, data exchange, and diagnostic. The first type includes the services `DDLM_Set_Prm`, `DDLM_Chk_Cfg` and `DDLM_Get_Cfg`, which are used to configure a slave prior to the periodic data exchange phase. The second type includes the services `DDLM_Data_Exchange` and `DDLM_Global_Control`, where the former allows for the exchange of data between a master and slave, and the latter permits the synchronisation of master and slave devices. Diagnostic services only include the `DDLM_Slave_Diag` service, which is used by masters to inquire slave stations about their state.

From the point of view of the user, DP operates asynchronously. During normal operation, the user of the DP AL only has to read or write data from a set of fixed memory positions, which represent the real value of variables used by a slave. The DP AL is responsible for reading the data from the slaves and placing it on their respective memory area. The data written by the user on the memory area is read by the DP protocol and written into the respective slave.

Configuration and parameterisation data are also stored at a specific memory area and, at start-up, the DP protocol uses this information to correctly configure the slaves according to the user specifications.

All these operations of the DP protocol are controlled by two state machines: the *slave state machine* and the *slave handler state machine*. The first is controlled by a slave whereas the second is used by a master to control the exchange of data with a specific slave. These are detailed next.

Slave State Machine

The s*lave state machine* controls the handling of DP protocol services by a slave. Figure 2.3 depicts a simplified state machine describing its behaviour. For further intuition, the reader is also referred to Figure 2.5, presented later in this sub-section, which illustrates the messages and services related to the evolution of the state machine.

On start-up, the slave's state machine goes into the `POWER_ON` state for internal initialisation purposes, and evolves into states `WAIT_PRM` and `WAIT_CFG` at the reception of the `DDLM_Slave_Diag.ind` and the `DDLM_Set_Prm.ind` service primitives, respectively. In these states, the slave configures itself using the information contained in the received indications.

The state machine evolves into the `DATA_EXCH` state upon reception of a `DDLM_Chk_Cfg.ind` primitive. In this state, the slave is able to exchange data with its controlling master. From the states described above, the slave only returns into the `WAIT_PRM` state when it detects a fatal error or aborts.

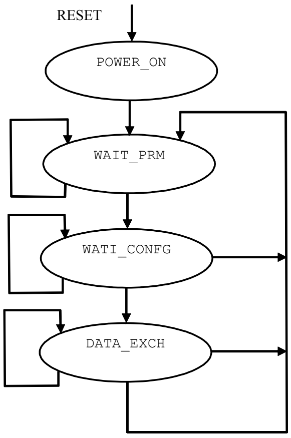


Figure 2.3 – *Slave state machine* of a DP slave (simplified)

Slave Handler State Machine

The *slave handler state machine* is used by a master to control the exchange of data with a specific slave. A PROFIBUS-DP master may control the evolution of several *slave handler state machines*, one for each slave.

Before entering into the data exchange phase, a master must send its configuration data to the slave. Only after confirming that all configuration data is correct, the master can evolve to the data exchange phase. Figure 2.4 and Figure 2.5 describe the operation of the master's *slave handler state machine*.

The *slave handler state machine* enters into the `STOP` state at power-on, initialises itself and goes into `DIAG1`, which calls the `DDLM_Slave_Diag.req` service primitive, with the objective of retrieving diagnostic information from the slave (e.g. the status of the slave and information about its channels). After receiving the confirmation to the `DDLM_Slave_Diag.req`, the state machine evolves to the `PRM` state if the confirmation reports a successful response from the slave, otherwise it returns to the `DIAG1` state.

In the `PRM` state, the master sends the parameters to the slave using the `DDLM_Set_Prm.req` service primitive. When a successful `DDLM_Set_Prm.con` is received, the state machine evolves to the `CFG` state, otherwise it returns to the `DIAG1` state.

In the `CFG` state, the master checks the actual configuration of the slave, using the `DDLM_Chk_Cfg.req` service primitive for that purpose. When a slave receives a `DDLM_Chk_Cfg.ind`, it checks its current configuration against the configuration contained in that message. If the master receives a successful confirmation, it evolves to state `DIAG2`, otherwise it returns to the `DIAG1` state.

During the `DIAG2` state, the master repeats the reading of the diagnostic data from the slave using the service primitive `DDLM_Slave_Diag.req`, waits for its confirmation and evolves to the data exchange phase, the `DATA` state.

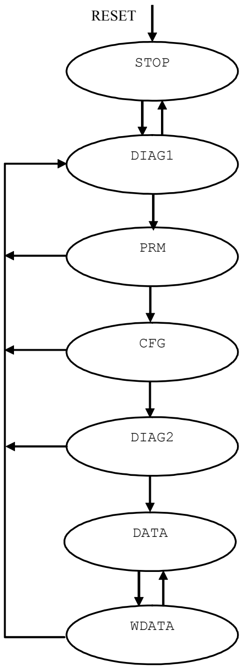


Figure 2.4 – ***Slave handler state machine*** **(simplified version)**

When on the DATA state, a master can exchange data with the slave, using the service primitive DDLM_Data_Exchange.req or send new parameter data to the slave using the service primitive DDLM_Set_Prm.req. In both cases, the state machine evolves to the WDATA state when a confirmation with status OK is received. The state machine also evolves to the WDATA state even when no answer is received from the slave or when the master is out of logical ring. This characteristic is the basis of the Inter-Domain Protocol (IDP), which will be described in detail in Chapter 6.

Additionally (neither described in Figure 2.4 nor in Figure 2.5), a master can also send the Sync and Freeze commands to the slaves or to a specific group of slaves. This command is sent using the service DDLM_Global_Control. When a slave receives the Sync command, its outputs are kept (frozen) in their current state. After, at the reception of another Sync command, the output state is updated with the new values transmitted meanwhile. The Freeze command is used to hold the state of the device input data on their current state, until the reception of another Freeze command. Also, during the time that elapses between the two commands, the controlling master can retrieve the current state of the slaves' input values.

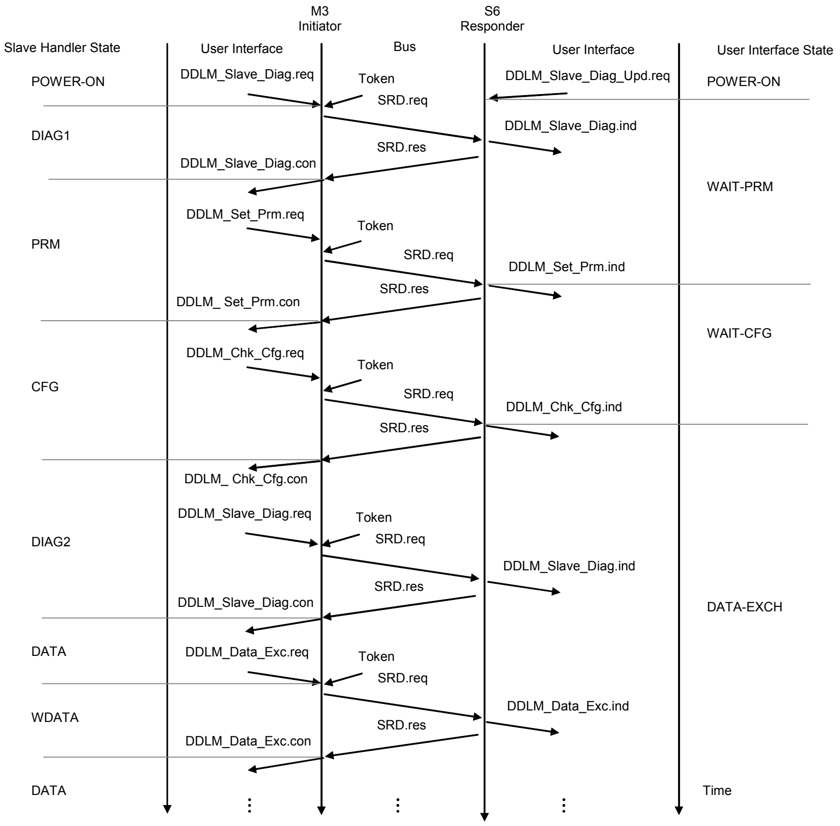


Figure 2.5 – Slave initialisation procedure scenario

The PROFIBUS-DP also defines some protective mechanisms, based on timers, both for masters and slaves. The masters use a timer for each *slave handler state machine* to control if they have retrieved data from the respective slave, otherwise the user is notified. Since PROFIBUS-DP does not define any particular name for this parameter, in the remainder of this thesis it will be referred as `Master_Watchdog_Timer`. As it will be highlighted later in this thesis, this parameter must be carefully set such as for the IDP does not generate errors. Slave stations also maintain a watchdog timer that is reset every time data is exchanged with its controlling master, otherwise the slaves' outputs enter into a failsafe state.

2.3. Relevant Details on RFieldbus

A RFieldbus (Alves *et al.*, 2002; Rauchhaupt, 2003) wired/wireless fieldbus network is composed by stations with a wireless interface that are able to communicate with wired (legacy) PROFIBUS stations.

The wireless part of the network includes at least one radio cell. A radio cell is a 3D-space where all associated wireless stations are able to communicate with each other. Taking into account that radio cells may be overlapping (usually it is intended), they must operate in different radio channels.

In RFieldbus, the interconnection between wired and wireless domains is made through Link Intermediate Systems (LIS) which relay frames at the PhL level.

Wireless communications in a radio cell may be achieved in two ways: in a direct way (*Ad-hoc* network) or via Base Station (Structured network).

In an *Ad-hoc* network configuration all stations in a cell inter-communicate directly, resulting that the coverage area of the wireless domain is equal to the interception of the radio coverage area of the individual stations. Figure 2.6 depicts a network scenario constituted by two wired PROFIBUS masters {M1 and M2}, one wireless PROFIBUS master {M3}, two wired PROFIBUS slaves {S1 and S2} and two Link Intermediate Systems {LIS1 and LIS2}. The radio coverage area of the wireless domain is determined by the interception of the radio coverage areas of each wireless device: {M3, LIS1, LIS2}.

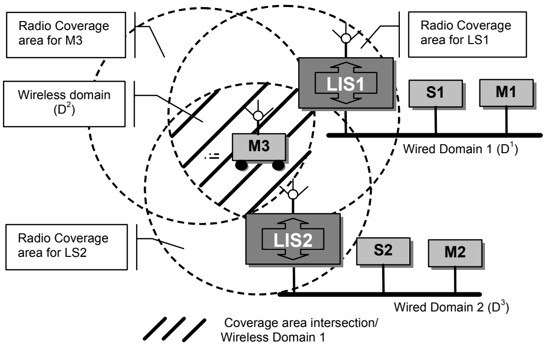

Radio Coverage area for M3
Radio Coverage area for LS1
Wireless domain (D^2)
LIS1
S1
M1
M3
Wired Domain 1 (D^1)
Radio Coverage area for LS2
LIS2
S2
M2
Wired Domain 2 (D^3)
Coverage area intersection/ Wireless Domain 1

Figure 2.6 – Example of a RFieldbus network with an *Ad-hoc* radio cell

In a *Structured* network configuration, all communications are relayed through a Base Station (BS). A BS operates as a wireless repeater using two radio channels, one to receive frames from the wireless stations (the uplink channel), and another to transmit frames to wireless stations (the downlink channel). The wireless domain coverage area is therefore defined by the coverage area of the BS (Figure 2.7).

In RFieldbus, the Link Intermediate Systems operate essentially as repeaters; that is, they receive frames from the wired domain, modify their PhL frame format and transmit those frames to the wireless domains, and vice-versa. Actually, the format of the wired and wireless PhL frames is different. The wireless frames include additional preamble and header fields. Additionally, each DLL character is coded for PhL transmission using 8 or 11 bit, for wireless and wired frames, respectively. One of the main characteristics of the RFieldbus approach is that it creates a “broadcast” network: the token rotates between all masters in the network and all communications are received by all stations in the network.

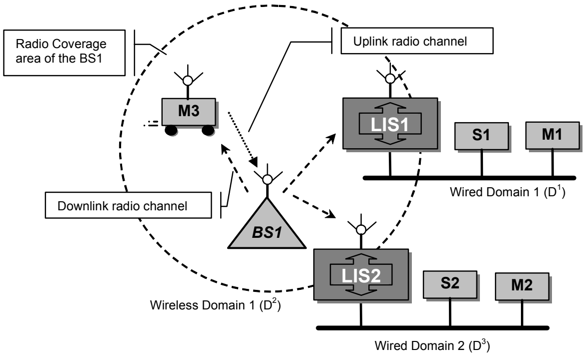


Figure 2.7 – Example of a RFieldbus network with an structured radio cell

Figure 2.8 illustrates an example where the wireless domains use a structured network configuration, and therefore the Link Intermediate Systems include the BS functionalities on their wireless front-ends. Such kind of devices is referred to as Structuring & Linking Intermediate Systems (SLIS).

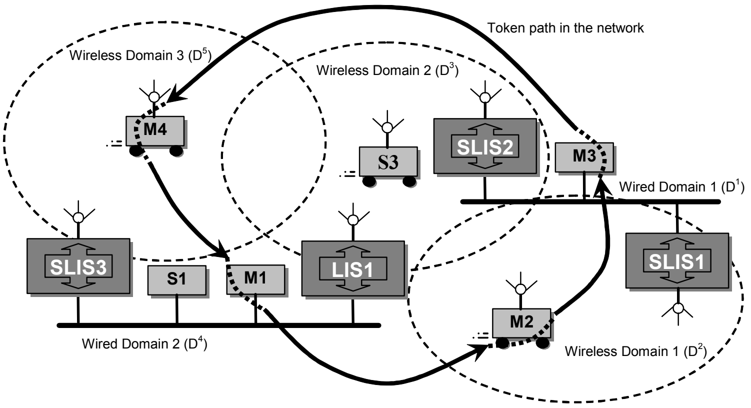


Figure 2.8 – Structured RFieldbus network with SLIS

2.3.1. Message Transactions and Basic Parameterization

In a RFieldbus system, the wired domains use the PROFIBUS PhL (RS-485 asynchronous version), whereas wireless domains are based on the IEEE 802.11b Direct Sequence Spread Spectrum (DSSS) PhL (Koulamas *et al.*, 2001a).

The wireless PhL operates at 2 Mbit/s, but requires the use of extra synchronisation and header fields. Additionally, each wireless character is coded using just 8 bits, while PROFIBUS RS-485 PhL codes a character using 11 bits. Consequently, the duration of the frames in wired and wireless domains is different. The result of this characteristic is that queuing delays may appear at the Link Intermediate Systems as outlined next.

Figure 2.9 depicts a simplified scenario for a hybrid network comprising two domains, one wired (D^1) and another wireless (D^2). The transactions take place between an initiator and a responder in domain D^1, and the LIS operates as a cut-through repeater. Due to the different bit rates and frame formats, the LIS can only start transmitting the frame after having received the byte containing the frame size, and after guaranteeing that the frame will be transmitted to the destination domains without gaps. That is the reason for the LIS delay represented in the figure.

In the scenario represented in the figure, there is an increasing queuing delay which is due to the duration of the frames in domain D^2 being higher than in domain D^1. The consequence of such behaviour is that the queuing delay may increase unboundedly (Alves, 2003) and in the case of the 4th transaction, between a master belonging to domain D^1 and a slave in domain D^2, the transaction duration (C_{req4}) is already significant.

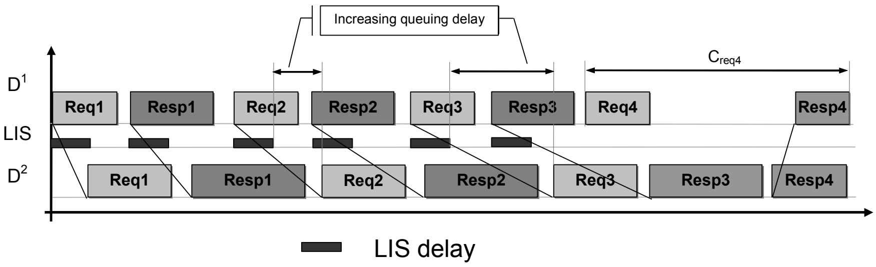


Figure 2.9 – Intermediate system queuing delay

A solution to the problem has been proposed in (Alves *et al.,* 2002), where the authors proposed a method which relies on the manipulation of the PROFIBUS `Idle Time` parameters, by inserting an additional idle time before a master starts the transmission of a request frame. In this way, it is guaranteed that the repeater queues do not increase in an undesirable way, compromising the real-time performance of the system. In (Alves, 2003), the methodology to properly compute the extra idle time to be inserted is described in detail. Figure 2.10 illustrates, for the same scenario of Figure 2.9, the simplified behaviour of the network when additional idle time (T_{ID1}) is inserted.

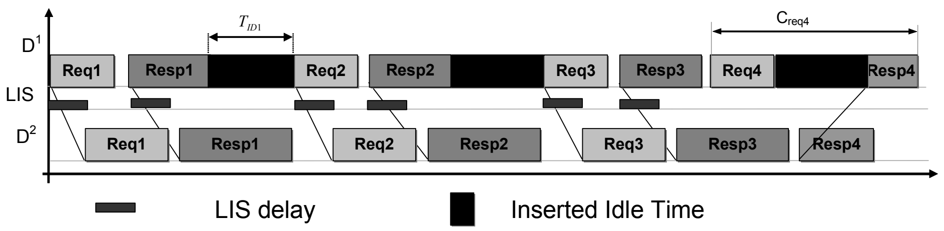


Figure 2.10 – Using inserted idle time for media adaptation

It can be observed in the figure that the duration of C_{req4} was reduced, due to the insertion of additional idle time.

Another consequence of the RFieldbus approach is that the setting of the `Slot Time` parameter must be made in accordance with the new values for the `Idle Time` parameter and the worst-case duration of message transactions. Figure 2.11 depicts a simplified example which illustrates the extra latencies in a message transaction due to the repeater-based approach. The total duration of a message cycle can be given by the following formulation:

$$C = C_{req} + t_{st} + C_{resp} + T_{ID1} \tag{2.5}$$

C_{req} is the duration of the request frame. C_{resp} is the duration of the response frame. t_{st} is the system turnaround time, which is equal to the time elapsed since an initiator ends the transmission of a request until it starts receiving the correspondent response, and can be computed by adding the system turnaround time without queuing delays (t_{stn}) and the worst-case queuing delay (Q). The reader is referred to (Alves, 2003) for further intuition and details.

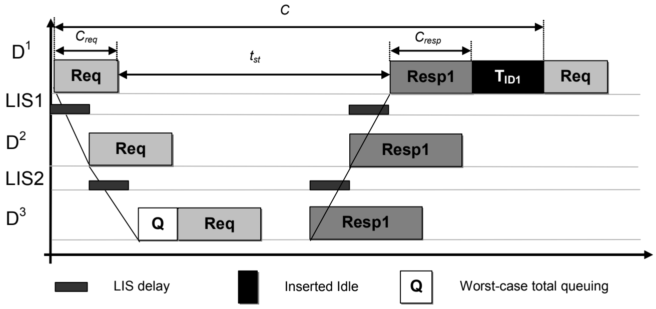


Figure 2.11 – Message transaction duration

Computing T_{SL1} (the `Slot Time` after the transmission of a request frame) and T_{SL2} (the `Slot Time` after the transmission of a token frame), depends on the system turnaround time after the transmission of a request (t_{st}) or after the transmission of the token (t_{st_token}) , respectively. The following formulation enables their calculation:

$$\begin{aligned} T_{SL1} &= \max\left\{t_{st}(S_i^k)\right\} \\ T_{SL2} &= \max\left\{t_{st\_token}(M)\right\} \end{aligned} \tag{2.6}$$

where $t_{st}(S_i^k)$ is a function of every i message stream S_i^k from master k, and $t_{st\_token}(M)$ is the a function of every master in the network. Them, $T_{SL} = max\{T_{SL1}, T_{SL2}\}$.

2.3.2. Supporting Inter-cell Mobility

In RFieldbus, mobility between different radio cells requires the use of structured radio cells. Therefore, all wireless frames are relayed through BSs.

The mobility mechanism is based on the role of a specific master station (the Mobility Master), which is responsible for periodically triggering the mobility management procedure. The Mobility Master starts the procedure by broadcasting a special frame – the `Beacon Trigger` (BT). The reception of this frame causes BSs to start transmitting `Beacon` frames in their radio channels and Mobile Stations to assess (using the `Beacon` frames transmitted by the BSs) the quality of all radio channels.

Figure 2.12 shows the simplified operation of the mobility mechanism considering the network scenario depicted in Figure 2.8, with M3 operating as the Mobility Master. The figure depicts the path of the `Beacon Trigger` frame through the network, the transmission of `Beacons` in each domain and the channel assessment procedure for mobile wireless master M4.

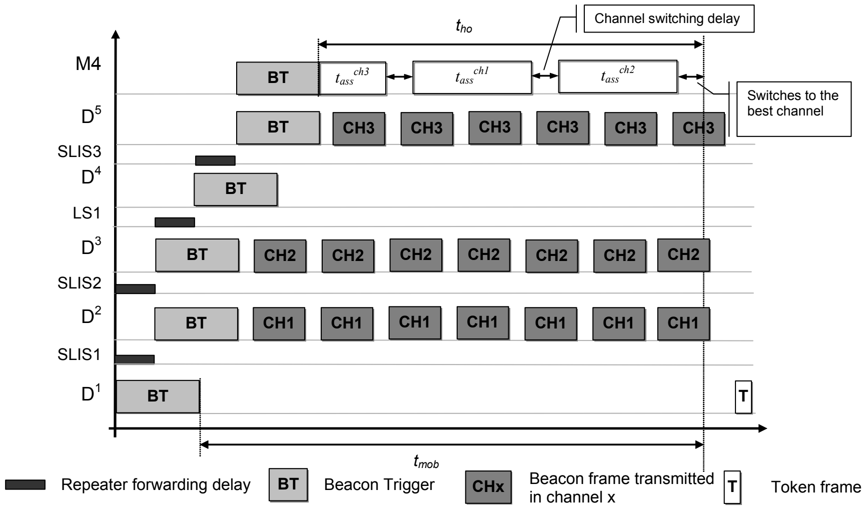


Figure 2.12 – Mobility Management procedure

M4 starts the handoff procedure immediately after the reception of the `Beacon Trigger` frame, it assesses its current channel (CH3), which requires a total duration of t_{ass}^{ch3} to be completed. Then, it proceeds likewise concerning CH1 and CH2. Finally, it switches to the radio channel with the best quality. The worst-case duration of the handoff procedure for M4 is equal to t_{ho}. During this process, the Mobility Master is idle, and consequently its `Idle Time` parameter (T_{ID2}) must be set in a way that guarantees that the mobile wireless stations in the systems are capable of assessing all wireless channels. At the end of this idle period, the Mobility Master passes the token to the next station in the logical ring.

Another consequence of supporting inter-cell mobility is that the setting of the `Slot Time` parameter must take into account the station location scenario that leads to the worst-case system turnaround time.

2.4. Some other Related Works and Technologies

Existing International Electrotechnical Commission (IEC) fieldbus standards (IEC61158 – Fieldbus Standard for use in Industrial Systems) rely on wired connections between devices, making them unsuitable for applications involving wireless mobile devices. Recently, several wireless LAN standards, like IEEE 802.11 (IEEE 802.11a, 1999; IEEE 802.11b, 1999), Bluetooth (IEEE 802.15.1, 2002) or Zigbee (IEEE 802.15.4, 2003; ZigBee, 2004) have appeared and could be used as a basis for the development of industrial wireless solutions.

However, industrial communication systems with wireless/mobility capabilities must fulfil the same basic requirements of traditional wired fieldbus networks. The problem arising from the use of commercially available wireless technologies is that they were not designed having industrial applications in mind. Therefore, most of the research work in this field addresses the adaptation or extension of standard communication protocols to guarantee real-time performance and high reliability levels.

One of the first contributions in this scope was the definition of a MAC mechanism based on a Time Division Multiple Access (TDMA) scheme to provide a wireless extension for WorldFIP (Morel *et al.*, 1995). In (Morel, 1995), the author suggested the use of Digital Enhanced Cordless Telecommunications (DECT) (ETSI standard: ETS 300 175, Parts 1 to 8) for supporting wireless communications between MAP/MMS nodes (where MAP is the acronym for Manufacturing Automation Protocol and MMS the acronym for Manufacturing Message Specification). In such an architecture, the mobility of wireless nodes is supported by the native mobility mechanisms offered by DECT.

(Cavalieri and Panno, 1997) addressed IEC/ISA (ISA is the acronym for the Instrumentation, Systems, and Automation Society) fieldbus wireless extensions by proposing a modified IEEE 802.11b protocol. Support for real-time communications in wireless domains uses the Point Coordination Function (PCF) offered by the IEEE 802.11b protocol. This functionality periodically creates a contention free period, which can be used to exchange periodic data without interference from the remaining traffic in the network. This approach very much resembles the Flexible Time Triggered operation, which is described in (Fonseca *et. al.*, 2000), albeit not in the wireless context.

In (El-Hoiydi and Dallemagne, 2000) the authors studied the behaviour of an IEEE 802.11 network and how its mobility mechanisms would affect the real-time traffic in the network. (Lee and Lee, 2001) proposed a wireless protocol based on the IEEE 802.11b MAC and PROFIBUS, in conjunction with a polling mechanism that ensures a deterministic performance. Nonetheless, the proposed mechanism requires changes to the PROFIBUS protocol operation. Additionally the proposed system does not support wireless PROFIBUS masters (only wireless slaves).

Willig analysed the capabilities of the PROFIBUS DLL together with the IEEE 802.11b PhL to support a wireless fieldbus network. In (Willig, 1999; Willig and Wolisz, 2001) the authors studied the ring stability of PROFIBUS over error prone links.

They performed field measurements on the Bit Error Rate (BER) in an industrial environment, and used this data for the development of a simulation model of a wireless PROFIBUS MAC. The authors concluded that with the BER levels encountered in most common modems (available at that time), PROFIBUS could be inadequate as a wireless protocol, since the probability of losing or corrupting a token on the wireless domains could be very high. Nevertheless, recent advances on wireless modems (like the REKA transceivers) enable a lower BER (Miaoudakis *et al.*, 2000) and consequently the probability of token loss to acceptable values. Notably, the REKA transceivers are used in RFieldbus. Also, in (Willig and Wolisz, 2001), the authors propose some changes to the PROFIBUS protocol and provide some guidelines which improve the operation of PROFIBUS ring management mechanisms in the presence of errors. Based on these findings, in (Willig, 2003) the author proposed and compared the use of polling-based communication algorithms with standard PROFIBUS, again in a wireless error prone environment, and concluded that these algorithms were capable of offering higher reliability characteristics.

A more recent work describes the extension of PROFIBUS-DP to operate over a Bluetooth wireless link (Miorandi and Vitturi, 2004a) and on a hybrid wired/wireless network supported by Ethernet and Bluetooth (Miorandi and Vitturi, 2004b). The proposed architecture maintains the compatibility at the Application Layer (AL) level but the PROFIBUS MAC sub-layer is replaced by the MAC sub-layer of Ethernet or Bluetooth protocols, which are provided with extensions that insure their real-time operation.

Currently, there are also some commercial solutions offering wireless extensions for fieldbuses. WaveCAN from Kvaser AB enables the interconnection of Controller Area Network (CAN) segments through a Bluetooth wireless link. It is interesting to point out that this solution is partially a result of the Mobile Fieldbus Devices in Industry (MOFDI) European project. The SATELLINE solution from SATEL supports the connection, by radio, of PROFIBUS devices at distances up to 50 Km. More recently, Siemens has also been offering a solution (SCALANCE W) based on an extension of IEEE 802.11b/g, capable of guaranteeing the real-time behaviour of a hybrid PROFINET network (IEC 61158-SER, 2005).

Besides RFieldbus, several research European-wide projects targeted wireless extension solutions for fieldbuses. One of the first was ESPRIT project 7210 – Open Low-Cost Time-Critical Wireless Fieldbus Architecture (OLCHFA) (Roberts, 1993) which developed a wireless extension for the WorldFIP fieldbus (IEC 61158-SER, 2005). This project was followed by MOFDI (ESPRIT 27035, Mobile Fieldbus Devices in Industry), already mentioned, that developed point-to-point wireless links based on the Bluetooth technology for CAN networks.

2.5. Summary

This chapter presented an overview of the PROFIBUS protocol and the RFieldbus system. The objective was to provide the reader with the necessary background and intuition for tackling the remainder chapters of this thesis. The chapter ended with a brief outlook on the most relevant research efforts for extending fieldbus technologies with wireless capabilities.

Chapter 3

Basics on PROFIBUS Timing Analysis

This chapter reviews some relevant previous research efforts for the provision of a worst-case response time analysis of PROFIBUS-based networks. Essentially, these results focus on traditional PROFIBUS networks, i. e. Single Logical Ring (SLR) systems. We then extend these results in order to consider the latencies associated with the logical ring management (masters joining the ring) and to the mobility mechanism in repeater-based systems (RFieldbus). These results will be basilar for the bridge-based timing analysis approach tackled in Chapters 7 and 8.

3.1. Introduction

A real-time computing system is defined as a system in which correctness depends not only on the logical result of the computation, but also on the time at which the results are produced (Stankovic, 1988). Therefore, a fieldbus network supporting real-time applications must not only guarantee that the communication between stations is reliable, but also that communications are timeliness.

In (Tovar and Vasques, 1999a), the authors have proposed the *Unconstrained Low-Priority Traffic Profile*, which enables the calculation of the Worst-Case Response Time (WCRT) of a message transaction in a Single Logical Ring (SLR) PROFIBUS network. An improvement of that work (Cavalieri *et al.*, 2002) proposed a less pessimistic approach which takes into account the high and low-priority traffic generated by all masters. These results, described in Section 3.2, are only directly applicable to a standard SLR PROFIBUS network. However, as it will be shown in Chapter 7, they constitute the basis upon which we will devise an analytical formulation for computing the WCRT of message transactions in a Multiple Logical Ring (MLR) network. This chapter also includes some important contributions to the timing analysis of SLR PROFIBUS networks (Section 3.3). One relates to the extension of the SLR analysis to consider the latencies associated to the logical ring maintenance (Section 3.3.2). This will be a crucial result for analysing the timing behaviour of the Inter-Domain Mobility Procedure (IDMP) proposed in Chapter 6.

Additionally, the analysis proposed by (Tovar and Vasques, 1999a) and (Cavalieri *et al.*, 2002) can give inaccurate results for the case of the repeater-based approach, since they do not take into account the duration of the inaccessibility period required by the mobility procedure and the behaviour of the Mobility Master. Therefore, in Section 3.3.3 we propose a new formulation for the calculation of the WCRT of a message transaction which takes into account the mobility mechanism used on the repeater-based approach and the behaviour of the Mobility Master. This result is important since it will enable a fair comparison between the SLR and MLR approaches, in Chapter 9.

3.2. Results Available for the Single Logical Ring Approach

In (Tovar and Vasques, 1999a), the authors suggest two different approaches to guarantee the real-time behaviour of a PROFIBUS-based system. In one of the approaches – the *Unconstrained Low-Priority Traffic Profile*, the real-time requirements for the high-priority traffic are satisfied, even when only one high-priority message is transmitted per token visit, independently of the low-priority traffic load. In this way, it is possible to have a guaranteed real-time approach for the high-priority message streams, provided that the relative deadline for these is higher than their worst-case response time ($Rslr_i^k$), which is given by:

$$Rslr_i^k = Q^k + Ch_i^k = nh^k \times T_{cycle}^k + Ch_i^k \tag{3.1}$$

In Eq. (3.1), nh^k is the number of high-priority message streams generated in master k. T^k_{cycle} is the worst-case token rotation time. Ch_i^k is the worst-case duration of a high-priority message cycle i issued by master k.

The exact characterisation of the cycle time properties of the PROFIBUS token is provided in (Tovar and Vasques, 1999b), which permits the evaluation the of T^k_{cycle} parameter in Eq. (3.1). An upper bound on the token cycle time can be given by:

$$T_{cycle}^k = T_{TR} + T_{del}^k = T_{TR} + n \times C_\sigma \tag{3.2}$$

T_{TR} is the PROFIBUS `Target Token Rotation` time parameter. n is the number of masters. C_σ is the longest message cycle in the network.

The above mentioned results can be somehow pessimistic, since it is considered that only one high-priority message cycle is performed per token visit and that low-priority messages are always present on the output queues. The analysis presented in (Cavalieri *et al.*, 2002) reduces some of this pessimism by taking into account the high and low-priority traffic generated by all masters in the network in every token cycle. The approach considers that the WCRT for a message stream from a master k is due to an initial blocking (B^k) caused by other masters with message transactions already going on, and due to the interference (I^k) caused by high-priority message streams (from master k and the other masters) and low-priority message streams (from other masters). The following notations are used to describe the approach followed in (Cavalieri *et al.*, 2002):

- k_i $\{k_0, k_1,\dots, k_{n-1}\}$ refers to the masters in the logical ring, where k_0 is the index for master k in the logical ring and $k_1,\dots, k_{n-1}$ are the i^{th} master receiving the token after master k_0;
- tc refers to the tc^{th} token visit to a master;
- $nh\pi_{tc}^{k_i}$ is the number of high-priority message cycles processed by master k_i in its tc^{th} token cycle;
- $\Delta h_{tc}^{k_i}$ is the value of the token holding timer at the token arrival to master k_i in its tc^{th} token cycle;
- $H_{tc}^{k_i}$ represents the time that master k_i spent processing message cycles during the tc^{th} token visit (note that this value can be higher then $\Delta h_{tc}^{k_i}$);
- $nl\pi_{tc}^{k_i}$ is the number of low-priority message cycles processed by master k_i in its tc^{th} token cycle;

- $\Delta l_{tc}^{k_i}$ is the time available for processing low-priority messages at the tc^{th} token cycle;
- $Ch_{max}^{k_i}$ is the longest high-priority message cycle performed by master $k_{i;}$
- $Cl_{max}^{k_i}$ is the longest low-priority message cycle performed by master $k_{i;}$
- λ is the total token latency.

The following equations allow the calculation of $H_{tc}^{k_i}$, $\Delta h_{tc}^{k_i}$ and $\Delta l_{tc}^{k_i}$:

$$\begin{aligned} H_{tc}^{k_i} &= Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \\ \Delta h_{tc}^{k_i} &= T_{TR} - \lambda - \sum_{i=0}^{n-1} H_{tc-1}^{k_i} \\ \Delta l_{tc}^{k_i} &= \Delta h_{tc}^{k_i} - nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} \end{aligned} \tag{3.3}$$

Obtaining the worst-case response time for high-priority message streams requires the simultaneous occurrence of the following worst-case conditions:

1. all high-priority requests are made at the critical instant, as defined in (Cavalieri *et al.*, 2002);
2. master k experiences the worst-case initial blocking;
3. it takes the maximum number of token visits (m) before all high-priority request are processed.

Based on these conditions, the worst-case response time ($Rslr^k$) for a message stream can now be re-written as follows:

$$R^k = Rslr_i^k = B^k + \sum_{tc=1}^{m-1} \left(H_{tc}^k + I_{tc}^k \right) + H_m^k + (m-1) \times \lambda \tag{3.4}$$

where m is given by:

$$m = \min \left\{ tc \in \aleph : \sum_{i=1}^{tc} nh\pi_i^k \geq nh^k \right\} \tag{3.5}$$

To obtain B^k, the following conditions which lead to the maximum initial blocking are considered to occur simultaneously:

1. all high and low-priority message streams, from master k, are simultaneously placed on the respective output queues just after master k releases the token;
2. no masters (including master k) use the token for processing message cycles in the token cycle prior to the critical instant in master k.

Given these worst-case conditions, B^k can be computed as follows:

$$B^k = \lambda + \sum_{i=1}^{n-1} \left(H_1^i \right) = \lambda + \sum_{i=1}^{n-1} \left(nh\pi_1^{k_i} \times Ch_{\max}^{k_i} + nl\pi_1^{k_i} \times Cl_{\max} \right) \tag{3.6}$$

where $nh\pi_{tc}^{k_i}$ is obtained as follows:

$$nh\pi_{tc}^{k_j} = \min\left\{ nh\rho_{tc}^{k_j} - \sum_{j=0}^{tc-1} nh\pi_{tc}^{k_j}, \left\lfloor \frac{\Delta h_{tc}^{k_j}}{Ch_{\max}^{k_j}} \right\rfloor^{+} + 1 \right\} \tag{3.7}$$

In Eq. (3.7), $nh\rho_{tc}^{k_i}$ represents the number of high-priority messages streams processed in master k_j, from the critical instant until the tc^{th} token visit to that master. This number of instances can be obtained as follows:

$$nh\rho_{tc}^{k_i} = \sum_{j=1}^{nl^{k_i}} \left\lfloor \frac{\lambda + \Theta_{tc} + \sum_{z=0}^{i-1}\left(H_{tc}^{k_z} + \frac{\lambda}{n} \right) + nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i}}{Th_j^{k_i}} \right\rfloor \tag{3.8}$$

where Θ_{tc} is the time span relative to master k, between the critical instant and the tc^{th} token cycle:

$$\Theta_{tc} = B + \sum_{m=1}^{tc-1}\sum_{i=0}^{n-1} H_{m}^{k_i} + (tc-1)\cdot\lambda \tag{3.9}$$

Going back to Eq. (3.6), $nl\pi_{tc}^{k_i}$ can be obtained by the following formulation:

$$nl\pi_{tc}^{k_i} = \begin{cases} 0 \text{ if } \Delta l_{tc}^{k_i} < 0 \\ \min\left(nl\rho_{tc}^{k_i} - \sum_{j=0}^{tc-1} nl\pi_{j}^{k_i}, \left\lfloor \frac{\Delta l_{tc}^{k_i}}{Ch_{\max}^{k_i}} \right\rfloor^{+} + 1 \right) \text{if } \Delta l_{tc}^{k_i} \geq 0 \end{cases} \tag{3.10}$$

where $nl\rho_{tc}^{k_i}$ is given by:

$$nl\rho_{tc}^{k_i} = nl^{k_i} + \sum_{j=1}^{nl^{k_i}} \left\lfloor \frac{\lambda + \Theta_{tc} + \sum_{z=0}^{i-1}\left(H_{tc}^{k_z} + \frac{\lambda}{n} \right) + nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} + nl\pi_{tc}^{k_i} \times Cl_{\max}^{k_i}}{Th_j^{k_i}} \right\rfloor \tag{3.11}$$

Finally, the interference (I_{tc}^{k}) that master k may suffer from message streams processed by other masters prior to the reception of the token by master k, is given by:

$$I_{tc}^{k} = \sum_{i=1}^{n-1} H_{tc}^{k_i} = \sum_{i=1}^{n-1}\left(Ch_{\max}^{k_i} \times nh\pi_{tc}^{k} + Cl_{\max}^{k} \times nl\pi_{tc}^{k} \right) \tag{3.12}$$

3.3. Some Contributions

3.3.1. Comments on the Results Presented in (Cavalieri et al., 2002)

The analysis proposed by (Cavalieri *et al.*, 2002) may, in some cases, lead to erroneous results. This is so because it considers that all transmitted messages have their length equal to the maximum length among all message streams. Therefore, the calculation of the token holding time ($H_{tc}^{k_i}$) can result not correct.

To illustrate the referred problem, consider a network with two masters operating with T_{TR} = 5.1 ms and λ = 0 ms. Consider also that master k_0 is transmitting a set of 5 message streams with a message cycle duration equal to {2 ms, 3 ms, 4 ms, 1 ms, 1 ms}.

Assume that master k_0 receives the token and that its $\Delta h_0^{k_0}$ value is equal to 5.1 ms, which is also equal to the value loaded into the T_{TH} timer. If the message streams are transmitted in the order presented before, the time that master k_0 spends processing message cycles during the 1st token visit is equal to 9 ms, i.e. master k_0 performs first a message cycle with a duration of 2 ms and then a message cycle with a duration of 3 ms. At this point, since the T_{TH} timer is equal to 0.1 ms, master k_0 can perform the message cycle with a duration of 4 ms, thus overrunning T_{TH} .

If the same calculations are made using Eq. (3.3), then the time that master k_0 spent processing message cycles during the 1st token visit is equal to 8.1 ms. In fact, Eq. (3.3) does not evaluate correctly the worst-case token holding time. Therefore we propose the following formulation to calculate the time that master k_0 spends processing message cycles during the tc^{th} token visit:

$$H_{tc}^{k_0} = \begin{cases} \Delta h_{tc}^{k} + Ch_{\max}^{k_i}, \text{if } \Delta h_{tc}^{k} < Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \\ Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i}, \text{if } \Delta h_{tc}^{k} > Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \end{cases} \tag{3.13}$$

For the other masters in the network, the number of low-priority messages transmitted must also be taken into account, therefore the correct formulation is proposed to be as follows:

$$H_{tc}^{k_i} = \begin{cases} \Delta h_{tc}^{k} + Ch_{\max}^{k_i}, \text{if } nl\pi_{tc}^{k_i} = 0 \text{ and } \Delta h_{tc}^{k} < Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \\ \Delta h_{tc}^{k} + Cl_{\max}^{k_i} \text{ if } nl\pi_{tc}^{k_i} > 0 \text{ and } \Delta h_{tc}^{k} < Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \\ Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i}, \text{if } \Delta h_{tc}^{k} > Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \end{cases} \tag{3.14}$$

3.3.2. Latencies Related to the Entry of a Master into the Logical Ring

The *Unconstrained Low-Priority Traffic Profile* assumes that only one high-priority message is transmitted per token visit. However, since the messages used by the Gap Update mechanism, the `FDL_Request_Status` frames, are low-priority messages, the *Unconstrained Low-Priority Traffic Profile* can only be applied to the Gap Update mechanism if the following conditions hold:

- a master *j* is in the `Listen_Token` state ready to enter into the logical ring in the gap interval controlled by master *k*;
- the previous GAP Update mechanism finished just before the time when master *j* is ready to enter into the logical ring;
- the high-priority message streams are all simultaneously queued in master *k* at the start of the GAP Update mechanism;
- the *`FDL_Request_Status`* frames are transmitted, on consecutive token cycles, after all high-priority message streams queued on master *k* are transmitted;
- all *`FDL_Request_Status`* frames start their transmission just prior to the expiration of the T_{TH} timer (as in the case of high-priority messages);
- the period of the high-priority message streams is higher than the duration of the Gap Update mechanism, consequently each message stream interferes only once during its duration.

Based on these assumptions, the worst-case time required by a master *j* to enter into the logical ring, in the GAP interval controlled by master *k* ($t_{master\_entry}{}^{k,j}$), can be computed as follows:

$$t^{k,j}_{master\_entry} = T_{GUD} + nh^{k} \times T^{k}_{cycle} + D_{k\to i} \times T^{k}_{cycle} + C_{FDL} + 2 \times \lambda / n \tag{3.15}$$

T_{GUD} is the `Gap Update` time, which is defined by PROFIBUS standard as a multiple (factor *G*) of $T_{TR} - T_{GUD} = G \times T_{TR}$. C_{FDL} is the latency of the `FDL_Request_Status` message and respective response. λ is the latency associated with the token passing between masters belonging to the logical ring, and *n* the number of masters in the network. Note that the PROFIBUS standard defines that if a master detects that its predecessor station has changed, it will only accept the token on its second try, and therefore the term $2\times\lambda/n$ is used in Eq. (3.15).

$D_{k\to j}$ is the distance parameter, which is defined as the number of addresses that master *k* must visit before inquiring station *j*. This quantity can be calculated by the following equation:

$$D_{k\to i} = \begin{cases} addr(j) - addr(k) - 1 & , addr(j) \geq addr(k) \\ HSA - addr(k) + addr(j) & , addr(j) < addr(k) \end{cases} \tag{3.16}$$

where *addr*(*x*) gives the numeric address of master *x* and *HSA* is the highest station address configured in master *k*. As mentioned in Chapter 2, in PROFIBUS an address can range from 0 to 126.

If several masters are added to the ring in the same GAP interval, then the time required until a master *j* enters the logical ring is given by the following equation:

$$t^{k,j}_{m\_master\_entry} = t^{k,m_1}_{master\_entry} + t^{m_1,m_2}_{master\_entry} + \ldots + t^{m_n,j}_{master\_entry} \tag{3.17}$$

where m_x represents the set of master stations that can enter into the logical ring in the same GAP interval as mobile master station *j*.

Considering the approach followed in (Cavalieri *et al.*, 2002), the conditions leading to the worst-case situation, regarding the entry of a master *j* into the logical ring, are similar to the conditions expressed in the beginning of this section:

- the previous GAP Update mechanism finished just before the time when master *j* is ready to enter into the logical ring;
- the high-priority message streams are all simultaneously queued in master *k* at the start of the GAP Update mechanism;
- the transmission of the `FDL_Request_Status` starts after the transmission of all high-priority messages and there is some available token holding time, during a token cycle, to perform it;
- just one `FDL_Request_Status` can be transmitted in a token cycle *tc*;
- the period of the high-priority message streams is higher than the duration of the GAP Update mechanism, consequently each message stream interferes only once during its duration.

To apply this analysis it necessary to consider another term ($\beta_{tc}^{k_i}$) in the calculation of $H_{tc}^{k_i}$, $\Delta h_{tc}^{k_i}$ and $\Delta l_{tc}^{k_i}$. $\beta_{tc}^{k_i}$ has a Boolean value depending on whether a `FDL_Request_Status` is transmitted during token cycle *tc* or not.

$$\begin{aligned}
&H_{tc}^{k_i} = Ch_{\max}^{k_i} \times nh\pi_{tc}^{k_i} + C_{FDL\_req} \times \beta_{tc}^{k_i} + Cl_{\max}^{k_i} \times nl\pi_{tc}^{k_i} \\
&\Delta h_{tc}^{k} = T_{TR} - \lambda - \sum_{i=0}^{n-1} H_{tc-1}^{k_i} \\
&\Delta l_{tc}^{k_i} = \Delta h_{tc}^{k_i} - nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} - C_{FDL\_req} \times \beta_{tc}^{k_i}
\end{aligned} \tag{3.18}$$

C_{FDL_req} represents the latencies associated with the transmission of a `FDL_Request_Status` frame, which does not receive any response from the addressed station. The actual value for $\beta_{tc}^{k_i}$ is:

$$\beta_{tc}^{k_i} = \begin{cases} 1 & \text{if } \Delta h_{tc}^{k_i} - nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} > 0 \\ 0 & \text{if } \Delta h_{tc}^{k_i} - nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} \le 0 \end{cases} \tag{3.19}$$

Since the GAP Update mechanism does not interfere with high-priority message streams, it is only necessary to reformulate the computation of the number of low-priority messages transmitted during token cycle *tc* – $nl\rho_{tc}^{k_i}$, in order to account for the `FDL_Request_Status` frames, as follows:

$$nl\rho_{tc}^{k_i} = nl^{k_i} + \sum_{j=1}^{nl^{k_i}} \left\lfloor \frac{\lambda + \Theta_{tc} + \sum_{z=0}^{i-1}\left(H_{tc}^{k_z} + \dfrac{\lambda}{n}\right) + nh\pi_{tc}^{k_i} \times Ch_{\max}^{k_i} + \beta_{tc}^{k_i} \times C_{FDL} + nl\pi_{tc}^{k_i} \times Cl_{\max}^{k_i}}{Th_j^{k_i}} \right\rfloor \tag{3.20}$$

Based on the formulation adapted from Eq. (3.18), Eq. (3.19) and Eq. (3.20), the following equation allows calculating the time required for a master *j* to enter into the logical ring in the GAP interval controlled by master *k*:

$$t_{master\_entry}^{k,j} = B^k + \sum_{tc=1}^{m-1}\left(H_{tc}^{k} + I_{tc}^{k}\right) + H_m^k + (m-1) \times \lambda + C_{FDL\_res} + 2 \times \lambda / n \tag{3.21}$$

C_{FDL_res} is the length of the FDL_Request_Status response. m is the of token cycles before master j has completed its entrance procedure into the logical ring. The remaining equation terms have the same meaning as in Eq. (3.3).

Finally, the stopping condition must be updated in order to guarantee that all GAP interval addresses until master j had been inquired:

$$m = tc : tc \in \aleph \; and \sum\nolimits_{z=1}^{tc} \beta_z^k = D_{j \to j} \tag{3.22}$$

3.3.3. Accounting for the Mobility Mechanism on the Repeater-based Approach

As explained in Chapter 2, in RFieldbus a Beacon Trigger message is transmitted by the Mobility Master, triggering the transmission of Beacons by the base stations. After transmitting the Beacon Trigger message, the Mobility Master becomes inactive for a duration of t_{mob}. That inactivity is accomplished by setting the Idle Time parameter (T_{ID2}) in the Mobility Master to a time equal to t_{mob}. As an example, on a network composed by 3 wireless radio cells (3 different radio channels) operating at 2 Mbit/s, this procedure may take approximately 2 ms (Alves *et al.*, 2002).

This means that, in practice, the Mobility Master will hold the token for a time longer than $T_{TR} + C_\sigma$. Since the mobility procedure is triggered periodically by the Mobility Master (depending on the speed of the movement of the mobile wireless stations), it is likely that no more than one mobility procedure is performed during the WCRT of a message transaction.

Therefore, the following equation updates Eq. (3.1) in order to account for the mobility procedure in RFieldbus system:

$$Rslr_i^k = (nh^k - 1) \times T_{cycle}^k + T_{cycle\_mob}^k + Ch_i^k \tag{3.23}$$

where $T_{cycle\_mob}{}^k$ is the duration of the token cycle time during the execution of the mobility procedure, and can be calculated by:

$$T_{cycle\_mob}^k = T_{TR} + (n-1) \times C_\sigma + t_{BT} + t_{mob} \tag{3.24}$$

In Eq. (3.24), $t_{BT} + t_{mob}$ is the worst-case token holding time of Mobility Master in which t_{BT} is the duration of the Beacon Trigger frame. Due to the need of setting a high value for T_{ID2} in the Mobility Master, it is advisable to use an independent Mobility Master, which only uses the network for the time required to periodically transmit the Beacon Trigger message and hold the token for the transmission of Beacons by the base stations. In such situation, it is possible to lower the pessimism of the analysis by considering only the mobility-related traffic on the Mobility Master. Therefore, the following equation updates the calculation of $T_{cycle}{}^k$ for the case when the Mobility Master is only used to transmit Beacon Trigger messages.

$$T_{cycle}^k = T_{TR} + (n-1) \times C_\sigma \tag{3.25}$$

In this specific case, the WCRT of message streams is still computed using Eq. (3.23).

For the approach which builds upon (Cavalieri *et al.*, 2002) work, taking into account the effects of the mobility procedure means considering the `Beacon Trigger` message as an additional high-priority message stream on the Mobility Master. This additional message stream (S_{mob}^{MM}) has the following duration and periodicity:

$$\begin{aligned} Ch_{mob}^{MM} &= t_{BT} + t_{mob} \\ T_{mob}^{MM} &= T_{per\_mob} \end{aligned} \tag{3.26}$$

where t_{per_mob} is the periodicity of the mobility procedure.

3.4. Summary

In this chapter, two alternative PROFIBUS timing analysis were presented which enable the calculation of the worst-case response time for a message stream in PROFIBUS SLR networks. The analysis proposed in (Tovar and Vasques, 1999a) is based on the *Unconstrained Low-Priority Traffic Profile*. It offers a simple formulation at the cost of a higher pessimism. The analysis proposed in (Cavalieri *et al.*, 2002) evolves that work by taking into account all high and low-priority traffic generated in the masters.

Both SLR approaches were extended in order to consider the latencies associated to the logical ring maintenance, specifically the entrance of a master into the logical ring, and to consider the inaccessibility period required by the mobility procedure and the behaviour of the Mobility Master.

Chapter 4

Aspects of the System Architecture, Network and Message Models

This chapter addresses the main characteristics of the proposed hybrid wired/wireless PROFIBUS bridge-based network. First, the generic issues related to the components of such a network are presented and characterised regarding their types and basic functionalities. Secondly, the main aspects of the bridge-based Intermediate Systems (IS) and the basics on the protocol extensions are presented. The chapter finalizes by presenting the analytical models of the proposed network and message streams.

4.1. Introduction

This thesis focuses on the use of bridges in a hybrid network architecture based on the PROFIBUS protocol. In this context, this chapter highlights the main aspects related to such a network architecture and its respective network and message models.

The chapter starts (Section 4.2) by characterising the different components of such a network. Such characterisation builds upon the classification proposed in (Alves, 2003). Then, Section 4.3 presents the fundamental aspects of the bridge-based ISs. Section 4.4 summarises the main issues related to the proposed protocol extensions that will be addressed in more detail later on in Chapters 5 and 6, namely on how Inter-Domain Transactions (IDT) and on how the Inter-Domain Mobility Procedure (IDMP) is implemented. Section 4.5 presents the analytical model of the proposed networking architecture and Section 4.6 presents an analytical model for the message streams. These analytical models focus on the characteristics that are relevant for the timing analysis which will be carried out in Chapters 7 and 8. Finally, Section 4.7 presents an instantiating example on the use of the proposed analytical models, with the propose of providing further intuition to the reader.

4.2. Basic Aspects of the System Architecture

The main components of the proposed communication network are the End Systems (ES) and the Intermediate Systems (IS). An ES is a communication node, supported by the PROFIBUS protocol, which contains the end-user applications. An IS is a device which interconnects different network domains.

A domain is defined as the set of ESs and ISs which can communicate among them through only one Physical medium. Domains can be of two types: wired and wireless.

A wired domain operates according to the PROFIBUS standard protocol.

The wireless part of the network is supposed to include at least one Radio Cell/(wireless domain). In the example illustrated in Figure 4.1, two wireless domains exist: D^1 and D^3. D^2 is a wired domain.

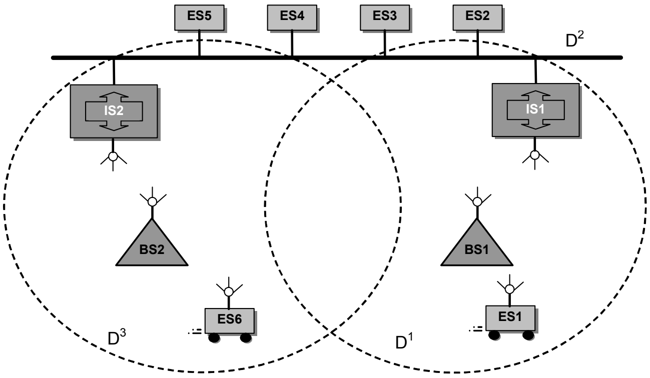


Figure 4.1 – Communication network basic example

Wireless domains can be of two types: *Ad-hoc* Wireless Domain (AWlD) or Structured Wireless Domain (SWlD).

In a SWlD all communications are relayed through a Base Station (BS). A BS is a type of IS capable of structuring a wireless domain. It operates as a repeater, receiving frames from the uplink channel and transmitting them on a downlink channel to the devices (ESs and ISs) which belong to the same wireless domain. In the remainder of this thesis, it is assumed that BS operate as cut-through repeaters with minimal latency.

In an *Ad-hoc* Wireless Domain (AWlD) the communication is made directly between the wireless ESs. In this type of wireless domain, inter-cell mobility is not supported.

4.2.1. Types of End Systems

Since the envisaged network is based on PROFIBUS, the ESs are classified according to the PROFIBUS type: master stations (M) or slave stations (S). In the remainder of this thesis, ESs are referred to as stations. Moreover, and for the sake of simplicity, master stations and slave stations will be simply referred to as masters and slaves, respectively.

Masters and slaves hold a communication stack complying with the PROFIBUS standard. A master is capable of issuing requests and belongs to a logical ring of token passing. A slave is a passive station only able to answer requests from masters.

The stations are also classified according to their communication medium – wired and wireless. In the example of Figure 4.2, the following set of wired PROFIBUS master and slave stations is considered: M1, M2, S1, S2, S3, S4 and S5. From these, the following are wireless stations: M3, S6 and S7.

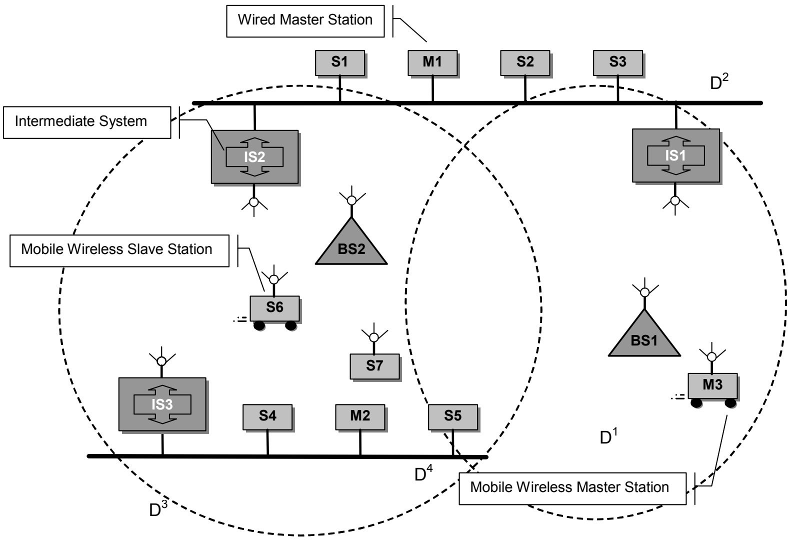


Figure 4.2 – A more detailed communication network example

A Wired Master/Slave Station is a master (WrM) or slave (WrS) which uses a wired medium for communication. A Wireless Master/Slave Station is a master (WlM) or slave (WlS) which uses a wireless medium for communication.

Within the set of wireless stations depicted on Figure 4.2, only M3 and S6 are mobile. S7 is not mobile or moves only inside a single wireless domain. Therefore, mobility is also a relevant characteristic of stations. A Mobile Wireless Master/Slave station is a wireless master (MWlM) or slave (MWlS) station capable of moving inside a wireless domain and between them.

A Domain Resident Wireless Master/Slave Station is a wireless master (DRWlM) or slave (DRWlS) station without inter-domain mobility capabilities. In the literature, Domain Resident Wireless stations are usually referred to as Stationary if they do not move at all.

4.2.2. Types of Intermediate Systems

The envisaged architecture may encompass four types of ISs: Linking Intermediate Systems (LIS), Structuring Intermediate Systems (SIS), Structuring & Linking Intermediate Systems (SLIS) and Mobile Linking Intermediate System (MLIS). These four types can be further classified into two groups. The Interconnecting Intermediate Systems group, constituted by Linking Intermediate Systems, Structuring & Linking Intermediate Systems and Mobile Linking Intermediate System, which are responsible for interconnecting two different media. The Structuring Intermediate Systems group is only responsible for structuring a wireless domain.

Linking Intermediate Systems

A Linking Intermediate System (LIS) interconnects a wired and a wireless domain (Structured or *Ad-hoc* Wireless Domains). It relays frames arriving from either side of the LIS to the other side, according to the routing information contained in the LIS (since bridge devices are considered in the architecture). Figure 4.3 depicts an example showing a Linking Intermediate System (LIS1) which interconnects wired domain D^2 with an *Ad-hoc* wireless domain D^1. The highlighted area represents the coverage area of the *Ad-hoc* wireless domain D^1.

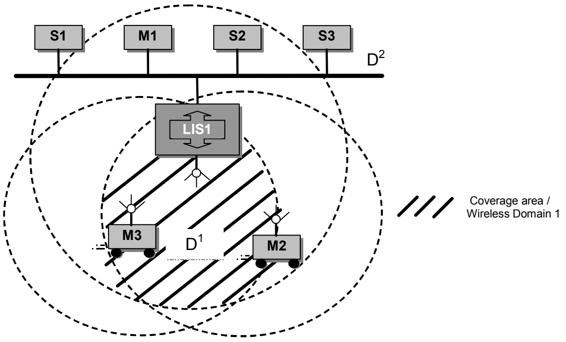


Figure 4.3 – Linking Intermediate Systems

Structuring Intermediate Systems

A Structuring Intermediate System (SIS) is used to interconnect wireless stations and Linking Intermediate Systems belonging to a structured wireless domain. Since all communications are relayed by the SIS, this device structures the wireless domain. Such a device operates by receiving frames transmitted in the uplink channel and re-transmitting the same frames on the downlink channel. It is important to note that just one SIS is allowed on a structured wireless domain. A SIS may be referred to as a Base Station (BS). Figure 4.4, depicts a network example illustrating the basic operation of a BS.

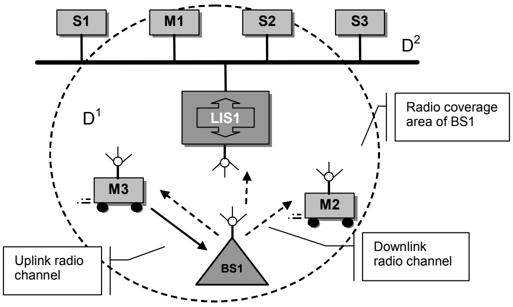


Figure 4.4 – Structuring Intermediate System

Structuring & Linking Intermediate Systems

A Structuring & Linking Intermediate System (SLIS) combines the functionalities of a LIS with the functionalities of a SIS into the same device. Therefore, such a device is capable of structuring a wireless domain and, at the same time, capable of relaying frames between the two domains. Figure 4.5 depicts an example where SLIS1 structures wireless domain D^1.

It should be noted that just one SLIS is allowed to connect to a wireless domain, while remaining ISs on the domain must be of the type Link Intermediate System or Mobile Linking Intermediate System (described next). A wired domain can connect to several SLIS, each one structuring a wireless domain.

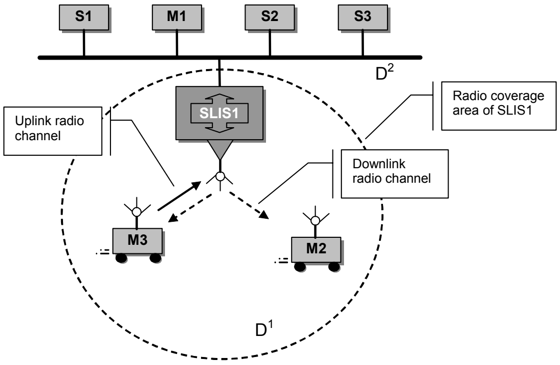


Figure 4.5 – Structuring & Linking Intermediate Systems

Mobile Linking Intermediate System

In some situations, it is advantageous that a set of stations are able to move together between wireless domains. For example, considering the scenario in Figure 4.6, if the stations belonging to domain D^4 are part of a mobile vehicle, then this set of stations is referred to as a Mobile Wired Domain.

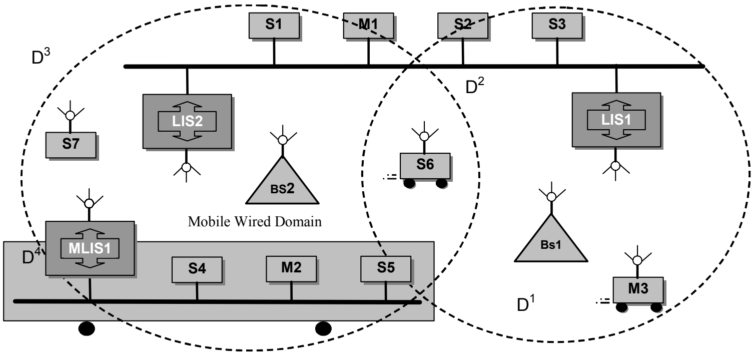


Figure 4.6 – Mobile Wired Domain

Therefore, a Mobile Wired Domain (MWrD) is defined as a set of stations that move physically together, between wireless domains. The connection with the other network domains is performed by Mobile Linking Intermediate Systems (MLIS), which must include inter-domain mobility support. The stations that connect to this type of domains are referred as Mobile Wired Masters (MWrM) or Mobile Wired Slaves (MWrS).

4.3. Basic Aspects of Bridge-based Intermediate Systems

On the envisaged network architecture, the Interconnecting Intermediate Systems operate as bridges. Therefore, frames exchanged between stations in different domains are relayed through bridges. A Bridge is a network device capable of relaying PROFIBUS DLL frames between the domains to which the bridge is connected. Although a bridge can interconnect more than two different domains, for the sake of simplicity and without loss of generality, in the remainder of this thesis it is assumed that a bridge only interconnects two different domains.

The bridge relaying decision is based on a routing table which determines whether an incoming frame is to be relayed to the other port or not. A bridge is constituted by two Bridge Masters (BM) and an interconnecting module.

Figure 4.7 presents a bridge (B1) of the Linking Intermediate System group type with two BMs {M1 and M2}. The bridge functionalities will be described in more detail in Chapter 5.

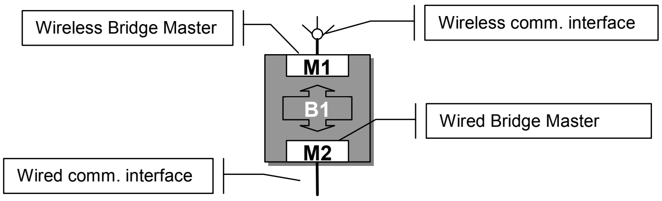


Figure 4.7 – Basic components of a bridge

If a bridge is also capable of structuring a wireless domain, then its wireless front-end must also implement the functionalities of a Structuring Intermediate System. Figure 4.8 depicts such kind of IS.

Figure 4.9 depicts, for the same network example illustrated in Figure 4.2, how such network is structured when bridge-based devices are used instead of repeater-based devices.

A BM is a PROFIBUS master modified in order to be capable of receiving all frames arriving to its physical interface, and forwarding them to the other BM of the bridge according to the routing information. These BMs operate almost as standard PROFIBUS masters and are assigned a PROFIBUS DLL address. Consequently, they take part of the logical ring of the domain to which they are connected. For the sake of simplicity, it is assumed that BMs do not support any AL functionalities.

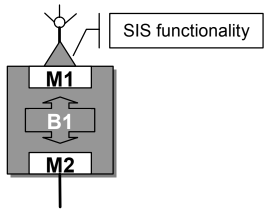


Figure 4.8 – Bridge including a Structuring Intermediate System front-end

BMs can be classified according to the medium to which they are attached to as: Wired BM and Wireless BM.

A Wired Bridge Master (WrBM) is a BM that connects a bridge directly to a wired medium.

A Wireless Bridge Master (WlBM) is a BM that connects a bridge to a wireless medium. The wireless medium can be either a Structured Wireless Domain or an *Ad-hoc* Wireless Domain, depending on whether a Structuring Intermediate System is associated to that domain.

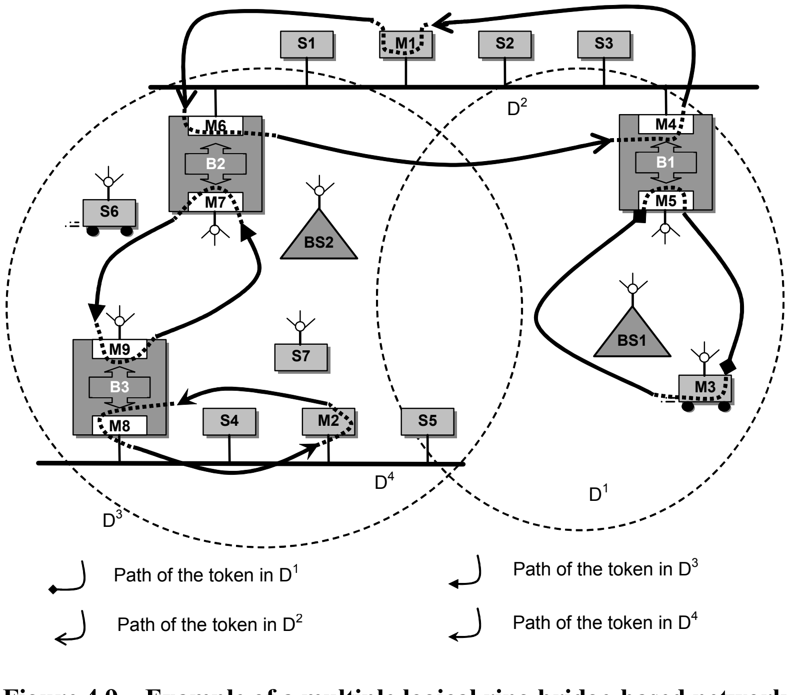


Figure 4.9 – Example of a multiple logical ring bridge-based network

4.4. Main Issues on the Protocol Extensions

This section summarises the extensions to the PROFIBUS protocol which are proposed in this thesis. The extensions are merely related to the proposed ISs.

4.4.1. Supporting Inter-Domain Transactions

Definitions and Concepts

In PROFIBUS, a message transaction involves a request by the initiator and an "immediate" response by the responder station. In a bridge-based network, when a transaction involves stations in two different domains, that sequence of events is not possible, since the request frame must be relayed by the bridge(s) until reaching the responder. Similarly, the response must be relayed by the bridge(s) until reaching the initiator. Thus, two types of transactions must be considered: Intra-Domain and Inter-Domain transactions.

An IntrA-Domain Transaction (IADT) is a transaction that involves stations in the same domain. In this case, the initiator and responder stations operate according to the rules defined by the standard PROFIBUS protocol.

An Inter-Domain Transaction (IDT) is a transaction which involves stations in different domains. In such type of transaction, the request and response frames are relayed by the bridge(s) and their respective BMs using a specific protocol: the Inter-Domain Protocol (IDP). The frames involved in IDTs (both the standard PROFIBUS frames and the frames exchanged between the BMs) are referred to as Inter-Domain Frames (IDF). IDFs conveying the request are called Inter-Domain Request (IDreq) frames and, equivalently, the frames which convey the response are called Inter-Domain Response (IDres) frames.

During an IDT, some BMs are involved in the transmission of the IDreq frames and on the reception of the response from the addressed slave (M4 and M7 in the case depicted in Figure 4.10). Other BMs are involved in the reception of the original request (issued by the initiator) and on the transmission of the Inter-Domain Response (M5 and M6 for the example depicted in Figure 4.10). Therefore, we define the IDT Communication Path as the set of stations involved in an IDT, which includes the initiator, the responder and the BMs which relay the request and the response. The IDT Communication Path starts on the initiator station.

The IDreq Communication Path is the ordered set of BMs involved in an IDT, which are responsible for transmitting the IDreq frames relative to an IDT to its destination domain. Note that when the initiator is a BM, then the first BM of the set is the first BM that transmits the request in a domain, which may not be the initiator. A BM can operate as a transaction initiator during the evolution of the IDMP.

The IDres Communication Path is the set of BMs involved in an IDT, which are responsible for transmitting the response frame relative to an IDT. The set is ordered starting from the first BM in the IDT Communication Path.

These notions will be used later on Chapter 7, for the definition of a mathematical model related to the worst-case response time of IDTs. Additionally, an example clarifying these concepts is presented later in Section 4.7.

The Inter-Domain Protocol

The IDP explores some PROFIBUS protocol features at the DLL and AL level, which enable a master to repeat the same request until receiving a response from the responder station. It defines the behaviour of the bridges and the codification of the frames exchanged between them, related to a specific IDT. While the IDP is defined in detail in Chapter 5, an overview of its operation follows.

When a master starts a transaction with a station belonging to another domain (an IDT), it starts by transmitting a request frame addressed to the responder station (an IDreq frame). This frame is then relayed by only one of the BMs (denoted as BM_{ini}) belonging to the initiator domain. BM_{ini} receives the IDreq, codes it according to the IDP (the coding of the frames is introduced later in Chapter 5), and stores internally information about the transaction. Meanwhile, the PROFIBUS DLL of the initiator retries transmitting the same request, since the BM_{ini} does not responds before the expiration of the `Slot Time` timer. The DLL retries are executed by the initiator a number of times specified by the `max_retry_limit` a DLL parameter.

The IDreq is relayed by bridges until reaching the last BM, which belongs to the responder domain. This BM is denoted as BM_{res}. BM_{res} decodes the original request frame and transmits it to the responder, which can be a standard PROFIBUS station (for example a wired PROFIBUS slave). When decoding the frame, the BM_{res} reconstructs the original frame as transmitted by the initiator (it even puts the initiator `Source Address` on the request frame). Thus, from the responder's perspective the initiator seems to belong to the same domain.

When the BM_{res} receives the response to that request, it codes that frame using the IDP and forwards it through the reverse path until reaching the BM_{ini}, where it will be decoded and properly stored.

Since for an IDT the response to the original request takes longer than for an IADT, the initiator AL must periodically repeat the same request until receiving the related response. After BM_{ini} having received (and stored) the correspondent response frame, it is ready to respond to a new (repeated) request from the initiator. The response frame is exactly equal to the frame transmitted by the IDT responder.

Considering the network scenario illustrated in Figure 4.9, Figure 4.10 represents a simplified timeline regarding a transaction between master M3 and slave S6. For the sake of simplicity, the retries issued by the DLL of the initiator are not represented. Note that DLL retries and AL repetitions are different concepts.

Figure 4.10 assumes the typical behaviour of PROFIBUS-DP, where the slaves read their inputs periodically, placing their image on the DLL by using the generic `Service_upd.req` primitive. The image of the input values is placed in a buffer, which is used by the DLL protocol to build a response to a specific request. An indication can be transmitted to the higher layers every time a slave receives a request. This type of procedure is usually referred to as buffered operation.

The initiator also uses a buffered communication mode, where the user and the initiator's protocol stack interface with the PROFIBUS-DP through a memory area, which allows reading and writing variables that represent the state of local or remote variables. It is the responsibility of PROFIBUS-DP to periodically update the variables using the DLL primitives of the type `Service.req`.

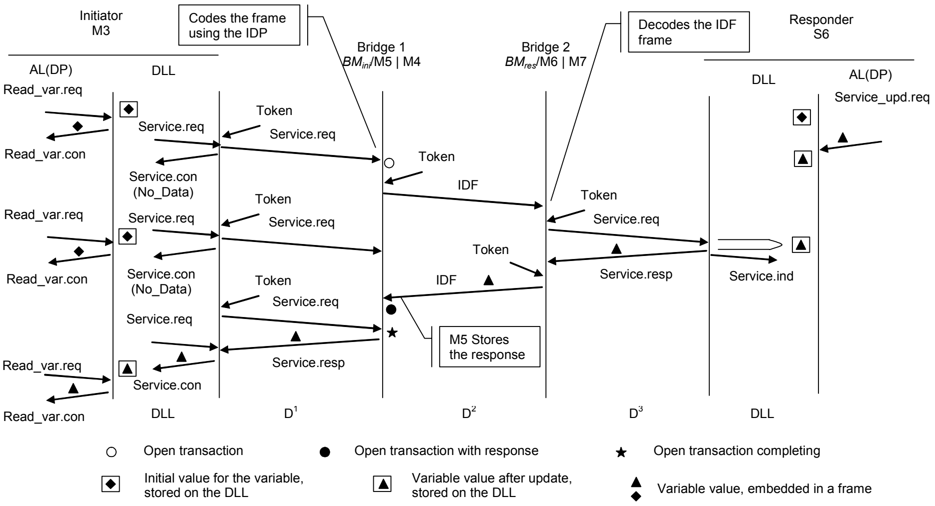


Figure 4.10 – Example of an Inter-Domain Transaction (IDT)

PROFIBUS-DP does not generate any errors to the AL level when no response is received since the PROFIBUS-DP *slave handler state machine* periodically repeats the same request, even if no answer is returned, until the expiration of the *Master Watchdog Timer* (at a master station). Consequently, if this timer is set to a value greater than the worst-case response time for an IDT, the user of the AL will not notice any errors, since a response will be received for one of the periodic repetitions of the request.

This operating mode guarantees compatibility between the IDP and the basic operation of PROFIBUS-DP, albeit some other compatibility issues which must be considered, as addressed later in Chapter 5.

4.4.2. Supporting Inter-Domain Mobility

One of the main requirements of the proposed architecture is to support inter-domain mobility. An Inter-Domain Mobility Procedure (IDMP) is proposed in this thesis offering transparent, reliable and real-time operation of the network in the presence of mobile stations or mobile domains. The IDMP is defined in a way that from the perspective of the PROFIBUS stations it is completely transparent whether the initiator, the responder, or both are moving between wireless domains.

The mechanism is based on a hierarchical structure, where one master, the Global Mobility Manager (GMM), is responsible for periodically starting the IDMP and controlling some of its phases. Additionally, in each domain, one master controls the mobility of stations belonging to that domain – the Domain Mobility Manager (DMM). The bridges also implement some specific mobility services. The mobile wireless masters and mobile wireless slaves implement specific services which enable them to evaluate the quality of the radio channels. These services are assumed to be similar to the ones used in the RFieldbus approach (described in Chapter 2).

Figure 4.11 shows the evolution of the IDMP, which is divided in four main phases.

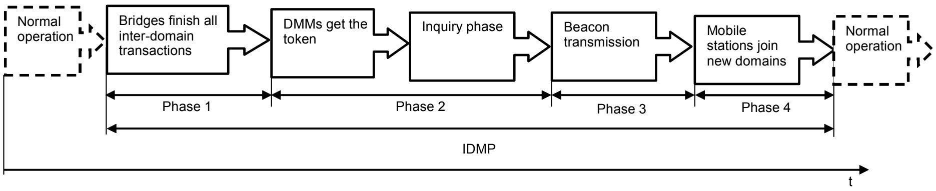


Figure 4.11 – Phases of the Inter-Domain Mobility Procedure (IDMP)

During Phase 1, the GMM commands all BMs to finish all pending IDTs (for which they are responsible). After receiving the confirmation that all DMMs finished their IDTs, the GMM starts Phase 2. During this phase, all DMMs are commanded to enter into the inquiry mode of operation (a sort of polling mode controlled by the domain DMM), during which only mobility related messages are exchanged. This type of operation allows a minimal latency for the communication between the GMM and the system DMMs, thus allowing a closer synchronization of the `Beacon` emission start instant.

The emission of `Beacons` is commanded by the GMM only after all DMMs end the inquiry sub-phase. The `Beacons` are used by the mobile wireless stations to evaluate the quality of adjacent radio channels. During phase 4, the DMMs of wireless domains try to detect which mobile wireless stations are present on their domains (using PROFIBUS ring maintenance mechanisms). If mobile stations are detected, the DMMs inform the bridges about their location. Chapter 6 details the operation of the IDMP.

4.5. Network Model

A hybrid wired/wireless PROFIBUS bridge-based network (*N*) is composed by a set of domains, stations (masters and slaves) and ISs:

$$N = (D, M, S, IS) \tag{4.1}$$

Table 4.1 depicts the parameters used to define a network (*N*).

Table 4.1 – Sets of components in a 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.
M	Set of PROFIBUS master stations in the Communication Network	$M = \{M^1,\ldots, M^{nm}\}$, where *nm* is the number of master End Systems in the Communication Network

Set	Description	Value
S	Set of PROFIBUS slave stations in the Communication Network	*S* = {S^1,…, S^{nsl}}, where *nsl* is the number of slave End Systems in the Communication Network
IS	Set of Intermediate Systems in the Communication Network	*IS* = {IS^1,…, IS^{ni}}, where *ni* is the number of ISs in the network

4.5.1. Model for the Communication Domains (D)

A communication domain is defined by a parameter set which contains its type, medium, ISs, stations (masters and slaves) and the set of PROFIBUS parameters common to all masters belonging to the domain:

$$D^i = \left(D\_TYPE, \omega^i, IS(D^i,t), M(D^i,t), S(D^i,t), D\_PPAR\right) : \mathrm{i} \in \{1\ldots\mathrm{nd}\} \tag{4.2}$$

with parameters as described in Table 4.2.

The functions described in the table return the network configuration at a certain time *t*. This feature is required to accommodate the different network configurations (due to station mobility) into the network model.

Table 4.2 – Communication Domain parameters

Parameter	Description	Value
D_TYPE	Represents the Communication Domain's type.	*D_TYPE* ∈ {WrD, SWlD, AWlD, MWrD} WrD – Wired Domain SWlD – Structured Wireless Domain AWlD – *Ad-hoc* Wireless Domain MWlD – Mobile Wired Domain
ω^i	Physical medium of *D*^{i}	
IS(D^i, t)	Function that returns the set of all ISs that are associated to *D^i* at time *t*.	
M(D^i, t)	Function that returns the set of all masters that are associated to *D^i* at time *t*.	
S(D^i, t)	Function that returns the set of all slaves that are associated to *D^i* at time *t*.	
D_PPAR	Set of PROFIBUS parameters which are common to all master stations on the domain	

PROFIBUS Parameters (`D_PPAR`*)*

The parameter `D_PPAR` represents the set of standardised PROFIBUS parameters, which are common to all masters in a specific domain:

$$D\_PPAR = \left(T_{SL}, G, HSA, max\_retry\_limit, T_{TR}\right) \tag{4.3}$$

These parameters are described in detail in Table 4.3. It is important to note that the PROFIBUS parameters together with the domain parameters reflect the possibility, given by the bridge-based network model, to operate with different settings in each domain.

As an example of this possibility, consider a situation in which a network domain operates at 12 Mbit/s and the remaining domains at 1.5 Mbit/s. Consider also that the wired domains have its Gap Update Factor (G) set to 10 and the wireless domains have that parameter set to 1.

Table 4.3 – Domain PROFIBUS parameters

Parameter	Description	Units
T_{SL}	Slot Time: time that an initiator waits for the receipt of a response/acknowledge after the transmission of a request.	bit time
G	Gap Update factor	-
HSA	Highest Station Address	-
max_retry_limit	maximum number of retries before giving up a request	-
T_{TR}	Target Token Rotation time	bit time

Model for the Physical Media (w)

A wireless physical medium usually differs from a wired one in the frame coding rules. In practice, these rules usually require the use of extra bit sequences in order to synchronise the emitter and receiver radio modems, and to allow error detection, correction or encryption. A physical medium (ω^j) is defined with the following set of parameters:

$$\omega^i = \left(r^i, lH^i, lT^i, k^i\right) \tag{4.4}$$

which are further detailed in Table 4.4.

Table 4.4 – Physical Media parameters

Parameter	Description	Units
r^i	Bit rate in physical medium i	Mbit/s
lH^i	Overhead of the frame head per PhL Protocol Data Unit (PDU) in physical medium i	bits
lT^i	Overhead of the frame tail per PhL PDU in physical medium i	bits
d^i	bits per DLL char for the PhL protocol of physical medium i	bits/char

The proposed model is a somewhat simplified version of the real physical behaviour, since the propagation delay is considered negligible and the bit error rate is not considered. Using these parameters together with the message stream parameters it is possible to calculate the duration of a physical layer frame (Alves, 2003) as follows:

$$t_{frame} = \frac{lH^i + L \cdot d^i + lT^i}{r^i} \tag{4.5}$$

where L represents the number of DLL characters of the frame.

4.5.2. Model for the Master Stations (M)

In the proposed bridge-based system, a master station is characterised by its type, mobility capabilities, message streams and PROFIBUS timing parameters as follows:

$$M^i = \left(ST\_TYPE, MOB\_FUNCT, St^i, nh, nl, M\_PPAR^i\right) : i \in \{1 \ldots nm\} \qquad (4.6)$$

for which details are given in Table 4.5.

Table 4.5 – Master parameters

Parameter	Description	Value
ST_TYPE	Represents the station's type	*ST_TYPE* ∈ {WrM, RWlM, MWlM, WrBM, WlBM} WrM – Wired Master DRWlM – Domain Resident Wireless Master MWlM – Mobile Wireless Master WrBM – Wired Bridge Master WlBM – Wireless Bridge Master
MOB_FUNCT	Represents the mobility functionalities which are supported by this station	*MOB_FUNCT* ∈ {MS, DMM, GMM, BR} MS – Mobile Wireless Station DMM – Domain Mobility Manager GMM – Global Mobility Manager BR – Bridge
St^i	Message stream set. This set can only be associated with WrM, DRWlM, MWlM	$St^b = \{S_j^1, \ldots, S_j^{ns}\}$, where *ns* is the number of Message Streams produced by M^b
nh	Number of high-priority message streams	$nh + nl \leq ns$
nl	Number of low-priority message streams	$nh + nl \leq ns$
M_PPAR^i	Set of PROFIBUS parameters related to a master station	

When a master M^i is a BM, *nh* and *nl* only reflect the maximum number of high and low-priority message streams being queued by that BM on a specific time instant in time. These values depend on the configuration of the network, concerning the location of the stations and the message streams conveyed by each one. They also depend on the state of the IDMP, which blocks IDTs during its evolution.

In Table 4.5, BMs are considered as master stations since they belong to the domain logical ring along with the other master stations. Additionally, a BM is involved in message transactions, together with the GMM and the domain bridges, related to the IDMP.

Although the role of the DMM or GMM could be performed by any non-mobile station, in the given model it is assumed that only the system BMs can perform such role, since only these devices are required to have some extensions to the PROFIBUS DLL.

Master PROFIBUS Parameters

The PROFIBUS standard defines a set of protocol related parameters for each station in a network. These parameters are used by the DLL to control the timings related to each transaction and to the timed token protocol.

The PROFIBUS protocol parameters for a master are defined as:

$$M\_PPAR = \left(ADDR, T_{RDY}, T_{SDI}, \min T_{SDR}, \max T_{SDR}, T_{SET}, T_{QUI}\right) \qquad (4.7)$$

Table 4.6 further details these parameters.

Table 4.6 – PROFIBUS master protocol parameters

Parameter	Description	Units
ADDR	Represents the PROFIBUS DLL address {0…126}	
T_{RDY}	Time within which a master station shall be ready to receive an acknowledgement or response after transmitting a request.	bit time
T_{SDI}	Station delay of the initiator, which is measured with respect to the receipt of the a frame last bit until an initiator is ready to transmit again.	bit time
min T_{SDR}	Minimum delay before a responder starts transmitting a response to a request.	bit time
max T_{SDR}	Maximum delay before a responder starts transmitting a response to a request.	bit time
T_{SET}	TSET is the set-up time which expires from the occurrence of an event (e.g. interrupt: last octet sent or Syn Time expired) until the necessary reaction is performed (e.g. to start Syn Time or to enable the receiver).	bit time
T_{QUI}	Transmitter fall time	bit time

4.5.3. Model for the Slave Stations (S)

In the proposed architecture, as in PROFIBUS, a slave station has a passive role, and is therefore defined by the following set of parameters:

$$S^{i} = \left(ST\_TYPE, MOB\_FUNCT, S\_PPAR^{i}\right) : i \in \{1 \dots nsl\} \qquad (4.8)$$

These parameters are further detailed in Table 4.7.

Table 4.7 – Slave parameters

Parameter	Description	Value
STATION_TYPE	Represents the station's type	*ST_TYPE* ∈ {WrS, RWlS, MWlS} WrS – Wired Slave RWlS – Resident Wireless Slave MWlS – Mobile Wireless Slave
MOB_FUNCT	Represents the mobility functionalities which are supported by this station.	*MOB_FUNCT* ∈ {MS} MS – Mobile Wireless Slave Station
S_PPAR^{i}	PROFIBUS parameters	

The operation of PROFIBUS slaves is much simpler than the operation of a master. It requires only the knowledge of the *min*T_{SDR} value, which represents the minimum time that a responder must wait before starting the transmission of a response, thus preventing the transmission of a response message when the initiator is not prepared. Additionally, a slave station is also required to have a PROFIBUS DLL address:

$$S\_PPAR = \left(ADDR, \min T_{SDR}\right) \tag{4.9}$$

4.5.4. Model for the Intermediate Systems (IS)

In the proposed architecture, the ISs generally operate as bridges. Without loss of generality, in this thesis, we are assuming that ISs contain only two BMs. The frames are relayed between BMs according to the information contained on the Routing Tables (RT) in each BM and, if necessary, are coded using the IDP protocol. This procedure entails a small processing delay which is measured from the reception of the last bit of a frame until the frame is placed on the other BMs output queue. This delay is called Internal Forwarding delay (ϕ).

ISs of the Structuring Intermediate System (or Base Station) type also require a small processing delay, due to its internal operation – the Internal Relaying Delay (Φ). For the sake of simplicity, the relaying delay is considered constant and independent of the frame size.

Therefore, an IS is modelled as follows:

$$IS^i = \left(IS\_TYPE, \Phi, \phi, BM, n_{beacons}\right) : i \in \{1 \ldots nis\} \tag{4.10}$$

with the parameters as described in Table 4.8.

Table 4.8 – Intermediate System parameters

Parameter	Description	Value
IS_TYPE	Type of IS, according to the IS classification and behaviour presented in Section 4.2.2	$IS\_TYPE \in$ {LIS, SIS, SLIS, MLIS} LIS – Linking Intermediate Systems SIS – Structuring Intermediate System SLIS – Structuring and Linking Intermediate Systems MLIS – Mobile Linking Intermediate System
Φ	Internal relaying delay (ms)	Only applies to IS of the type: {SIS, SLIS}
ϕ	Internal forwarding delay (ms)	Only applies to IS of the type: {LIS, SLIS, MLIS}
BM	Set of BMs which constitute the bridge i: BM^d={M^1, M^2}	
n_{beacon}	Number of `Beacons` transmitted by a SIS or by a bridge of the type SLIS.	

Network Topology and Address Space

The typical target industrial applications will not usually require the use of complex topologies (Behaeghel *et al.* 2003). The network topology is assumed to be tree-like and without loops between an initiator and a responder. This restriction avoids the need for more sophisticate network protocols to support routing.

In a tree-like topology, the domains which only have one connection with another domain are referred to as Terminal Domains. This notion is especially important for the mobility procedure, since these domains are not required to take some of the actions related to the evolution of the IDMP.

The network addressing is based on the available MAC addresses provided by PROFIBUS, creating a single address space. In the network it is possible to address at most 127 stations, with addresses ranging from 0 to 126 (address 127 is reserved for broadcasts).

Routing is based on a table contained in the bridges, which specify the MAC addresses of the stations on either sides of the bridge.

The network topology may change dynamically due to the inter-domain mobility of mobile wireless stations and mobile wired domains.

4.6. Message Stream Model (S_i^k)

A Message Stream is a periodic sequence of message transactions, related with a specific system functionality, e.g. the reading of a sensor. In PROFIBUS a transaction usually involves the sending of a request frame and the reading of a response frame, when the `Send and Request Data` (SRD) or the `Send Data with Acknowledge` (SDA) services are used. In the case a request is transmitted in unicast or broadcast mode, the initiator does not expect any response; that is the case of the `Send Data without Acknowledge` (SDN) service.

In Chapter 3, a model for a message stream has already been presented. In that model, which is traditionally used in this field of research, a message stream is referred to as S_i^k, where *k* represents the master (initiator for the message stream transactions) and *i* the stream index on master *k*. S_i^k represents the triplet $\{T_i^k, C_i^k, P_i^k\}$, where T_i^k is the message stream period, C_i^k the duration of a message transaction, P_i^k represents the priority of the message stream.

The message stream model presented below extends the "traditional" model by considering also, in the case of IDTs, the IDT communication path, and the different locations of the initiator and responder. The extended model for a message stream is thus defined as:

$$S_i^k\left[\left(D^{ini}\left[,D^{resp}\right]\right)\right]=\left(RESPONDER, L_{req}, L_{resp}, T_i^k, \Omega_{req}\left(S_i^k\right), \Omega_{resp}\left(S_i^k\right), P_i^k\right) \tag{4.11}$$

D^{ini} represents the domain where the initiator is located, and D^{resp} represents the domain where the responder is located. These parameters are optional, and are especially useful for the case when the message stream initiator and/or the responder are mobile wireless stations. The remaining parameters are as described in Table 4.9.

Since mobile wireless stations can move between different wireless domains, the communication path between stations changes in time. Thus, $\Omega_{req}(S_i^k)$ and $\Omega_{res}(S_i^k)$ only reflect the path on the network configuration specified by D^{ini} and D^{resp}. The result of these functions are vectors Ω_{req} and Ω_{res}. These vectors can have at most *b* elements, where *b* is the number of bridges on the IDreq communication path or the IDres communication path.

On a bridge-based approach, a message stream is equivalent to a set of different message transactions, at least one transaction for each domain in the IDT Communication Path. Combining the parameters of the message stream (S_i^k), the parameters for each domain (D^a), and the initiator and responder parameters, it is possible to calculate the duration of a message transaction in each domain.

Table 4.9 – Message stream parameters

Parameter	Description	Units
RESPONDER	The slave that is the responder for the transaction *RESPONDER* $\in$ $\{S^1,\ldots, S^{nsl}\}$	-
L_{req}	Length of the PROFIBUS standard DLL request message.	chars
L_{resp}	Length of the PROFIBUS standard DLL response message.	chars
T_i^k	Minimum periodicity.	ms
$\Omega_{req}(S_i^k)$	IDTreq Communication path for message stream	-
$\Omega_{res}(S_i^k)$	IDTres Communication path for message stream	-
P_i^k	Priority. $P \in$ {high, low}	-

The worst-case duration of a complete message transaction (SDR service), involving an initiator *k* and relative to message stream *i* (C_i^k), measured from the start of the request frame until the time when the initiator can start transmitting a new frame, is given by:

$$C_i^k = treq_i^k + \max T_{SDR} + tres_i^k + T_{ID1} + \Phi \tag{4.12}$$

where $treq_i^k$ and $tres_i^k$ represent the time required to transmit the request and response frames, respectively. These two latencies can be calculated using Eq. (4.5). Φ represents the relaying delay imposed on a structured domain by the structuring devices (a SLIS or a SIS). Obviously, in an *Ad-hoc* domain the value of Φ is equal to zero.

The duration of a transaction based on the SDN service, also measured from the start of the request frame until the time when the initiator can start transmitting a new frame, is given by the following formulation:

$$C_i^k = treq_i^k + T_{ID2} + \Phi \tag{4.13}$$

4.7. Instantiating the Network and Message Models to an Example

Figure 4.12 depicts an example scenario that will be used to instantiate the models described in this chapter, which can be represented according to Eq. (4.1) as follows:

$$N = (\{D^1, D^2, D^3, D^4, D^5\}, \{M1, M2, M3, M4, M5, M6, M7, \ldots \\ M8, M9, M10, M11\}, \{S1, S2, S3, S4, S5, S6\}, \{B1, B2, B3, B4\}) \tag{4.14}$$

The network contains 5 different domains $\{D^1, D^2, D^3, D^4, D^5\}$, two of which are wired domains (D^2 and D^3), two are structured wireless domains (D^1 and D^4), and one is a mobile wired domain (D^5).

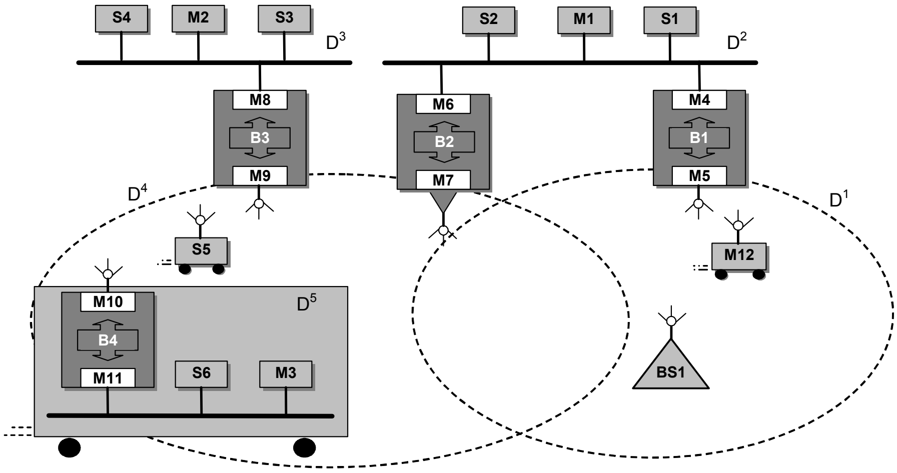


Figure 4.12 – Example scenario

Table 4.10 defines the domains in detail, according to the parameters specified in Eq. (4.2).

Table 4.10 – Domain parameters

Domain	Parameters
D^1	(SWlD, ω^1, {B1, BS1}, {M5, M12}, -, D_PPAR^1)
D^2	(WrD, ω^2, {B1, B2}, {M1, M4, M6}, {S1, S2}, D_PPAR^2)
D^3	(WrD, ω^2, {B2, B3}, {M2, M8}, {S3, S4}, D_PPAR^3)
D^4	(SWlD, ω^1,{B2, B3, B4}, {M7, M9, M10}, S5, D_PPAR^4)
D^5	(MWlD, ω^3,{B4}, {M3, M11}, S6, D_PPAR^5)

Note that this example explores the feature provided by the systems model which permits that the domains may have different physical media and PROFIBUS parameters. Table 4.11 presents the common PROFIBUS parameters for the stations (Eq. (4.3)).

Table 4.11 – Domain PROFIBUS parameters

Domain	Parameters
D_PPAR^1	(115, 5, 5, 1, 1, 300)
D_PPAR^2	(115, 5, 6, 1, 1, 300)
D_PPAR^3	(115, 5, 9, 1, 1, 300)
D_PPAR^4	(115, 5, 8, 1, 1, 300)
D_PPAR^5	(60, 5, 11, 1, 300)

Note that, in order to reduce the effect of the GAP update mechanism, each domain is set with a different value for the HSA. Table 4.12 depicts the values related to the different physical media according to Eq. (4.4). Note the different bit rates and also the different frame formats for wired and wireless domains. In a wired domain, a frame is coded using 11 bits per character. In a wireless domain, a frame is coded with just 8 bits per character added to 32 head and 16 tail bits.

Table 4.12 – Physical media parameters

Domain	Parameters
ω1	(2.0, 32, 16, 8)
ω2	(1.5, 0, 0, 11)
ω3	(0.187, 0 ,0, 11)

The set of master and slave stations is presented in Table 4.13. This table defines the type of stations, their mobility capabilities (in the case of BMs also their role on the IDMP) and PROFIBUS parameters. For the case of master stations, Table 4.13 also presents their set of message streams and number of high and low-priority ones. These parameters have been specified by Eq. (4.6) and Eq. (4.8), for a master and a slave, respectively.

Table 4.13 – Master and slave parameters

Master	Parameters	Slave	Parameters
M1	(WrM, -, St^{M1}, 2, 1, -, M_PPAR^{M1})	S1	(WrS, -, S_PPAR^{S1})
M2	(WrM, -, St^{M2}, 1, 0, -, M_PPAR^{M2})	S2	(WrS, -, S_PPAR^{S2})
M3	(WrM, -, St^{M3}, 1, 1, -, M_PPAR^{M3})	S3	(WrS, -, S_PPAR^{S3})
M4	(WrBM, -, -, -, -, -, M_PPAR^{M4})	S4	(WrS, -, S_PPAR^{S4})
M5	(WlBM, DMM, -, -, -, *100,* M_PPAR^{M5})	S5	(MWlS, MS, S_PPAR^{S5})
M6	(WrBM, GMM, -, -, -, -, M_PPAR^{M6})	S6	(WrS, -, S_PPAR^{S6})
M7	(WlSBM, DMM, -, -, -, *100,* M_PPAR^{M7})		
M8	(WrBM, DMM, -, -, -, -, M_PPAR^{M8})		
M9	(WlBM, -, -, -, -, *100,* M_PPAR^{M9})		
M10	(WlBM, -, -, -, -, -, M_PPAR^{1})		
M11	(WrBM, DMM, -, -, -, -, M_PPAR^{M10})		
M12	(MWlM, MS, St^{M12}, 1, 0, -, M_PPAR^{M11})		

Table 4.14 defines the set of master and slave PROFIBUS parameters, which were defined by Eq. (4.7) and Eq. (4.9), respectively. These parameters differ from each other since the bit rate in D4 requires the setting of its masters' differently from masters belonging to other domains. Note that these parameters have been chosen according to the PROFIBUS standard.

Table 4.14 – Master and Slave PROFIBUS parameters

Station	Parameters
$M\_PPAR^{\{M1,\ M3\text{-}M10,\ M12\}}$	({1, 3-10, 12}, 10, 100, 11, 100, 1)
$M\_PPAR^{\{M3,\ M11\}}$	({2, 11}, 10, 20, 11, 40, 1)
$S\_PPAR^{S1\text{-}S6}$	({1-6}, 11)

The following set of bridge ISs (Eq. (4.10)) interconnect the different network domains:

Table 4.15 – Intermediate System parameters

IS	Parameters
B1	(LIS, 0.030, {M4, M5}, -)
B2	(SLIS, 0.030, {M6, M7}, -)

IS	Parameters
B3	(LIS, 0.030, {M8, M9})
B4	(MLIS, 0.030, {M10, M11}, -)
BS1	(SIS, 0.030, -, -)

While the message stream model is defined by Eq. (4.11), Table 4.16 instantiates the set of message streams belonging to the master stations of the example network. Note that in the case of message stream S_1^{M12}, the IDreq and IDres communication paths are related to the case when M12 (a mobile wireless master) is located in domain D^1.

Table 4.16 – Message Streams for the example

Stream	Parameters	Stream	Parameters
S_1^{M1}	(S4, 20, 11, 5, {M7, M8}, {M6, M9}, high)	S_1^{M3}	(S3, 20, 11, 5, {M10, M8}, {M11, M9}, high)
S_2^{M1}	(S2, 13, 11, 5, -, -, high)	S_2^{M3}	(S3, 20, 11, 5, {M10, M8}, {M11, M9}, low)
S_3^{M1}	(S6, 20, 11, 5, {M7, M11}, {M6, M10}, low)	$S_1^{M12}(D^1)$	(S6, 20, 11, 5, {M4, M7, M11}, {M5, M6, M10}, high)
S_1^{M2}	(S4, 20, 11, 5, -, -, high)		

To calculate the maximum number of high-priority message streams that can be simultaneously queued by the system BMs it is necessary to consider all possible locations of M12. Table 4.17 resumes the results of those calculations.

Table 4.17 – Number of message streams relayed by the BMs

BM	nh	BM	nh
M4	1	M5	0
M6	1	M7	3
M8	3	M9	3
M10	4	M11	1

4.8. Summary

This chapter discussed the main characteristics of the proposed hybrid wired/wireless PROFIBUS bridge-based network, introducing the fundamental aspects of the system architecture and respective novel components. It is also introduced the protocol extensions required by the bridge-based operation. These protocol extensions are the IDP and the IDMP, which will be described in detail in the next two chapters. Analytical models for both network and message streams were also proposed. These will be crucial tools for the timing analysis proposed in Chapters 7 to 9.

Chapter 5

The Inter-Domain Protocol (IDP)

The PROFIBUS protocol has been originally designed for supporting communications in a “broadcast” wired network. Enabling the operation of PROFIBUS in a bridge-based network requires the specification of protocol add-ons, enabling transparent communications between stations belonging to different network domains. For this purpose, this chapter presents the details related to the Inter-Domain Protocol (IDP) by defining the architecture of the bridges and the operation of its main components.

5.1. Introduction

In PROFIBUS, a message cycle comprises a request frame sent by the initiator and an immediate response frame sent by the responder. This immediate response is obviously not possible for Inter-Domain Transactions (IDT), for which a solution relies on the provision of an Inter-Domain Protocol (IDP) associated with the bridge devices as briefly described earlier in Chapter 4.

The IDP is based on bridge devices which are constituted by Bridge Masters (BMs). These devices are modified PROFIBUS masters (at the Data Link Layer (DLL) level) with additional components required for the handling of IDP and of the Inter-Domain Mobility Procedure (IDMP). The architecture and components of bridge devices is described in Section 5.2.

In the proposed hybrid wired/wireless architecture the routing is made based on a single address space composed of standard PROFIBUS MAC addresses. Therefore, each BM has a Routing Table (RT) which contains information about the location of the network stations, enabling the routing operations. The details regarding the routing tables are described in Section 5.2.2.

The handling of IDTs requires that the transaction BM_{ini} must store information about the parameters of the ongoing IDTs. To fulfill that mission, the BMs store that information in a table called the List of Open Transactions (LOT), which contains information such as the IDT `Source Address` (SA), `Destination Address` (DA) and IDT state. Section 5.2.3 describes the operation of the LOT.

The frames exchanged by the IDP between bridges must contain information which allows reconstructing the original frame and matching the information contained in the BM_{ini} LOT with the respective response. To fulfill these objectives, the frames exchanged between the bridges in an IDT must follow specific IDP coding rules described in detail in Section 5.2.4.

In the proposed network architecture, the completion of an IDT depends not only on the success or failure of the transmission of the request between the initiator and the BM_{ini}, but also on the success or failure of the remaining transmissions between the stations relaying the IDFs related to the IDT. Although PROFIBUS already provides some error control mechanisms, these are not enough to prevent the occurrence of errors during the execution of IDTs. Therefore, Section 5.2.5 proposes the error control procedures required for the handling of errors during the execution of IDTs.

One of the main requirements of the IDP was the compatibility with legacy PROFIBUS stations. As a consequence, the IDP requires additional functionalities provided by the BMs to be capable of handling the initialization of PROFIBUS-DP slaves. These issues and other compatibility problems are discussed in Section 5.3.

Details regarding the implementation of the protocol are discussed in Section 5.4. Finally, Section 5.5 describes in detail the timings of an IDT associated to the network scenario.

5.2. Bridge Architecture

A bridge must include one BM for each of its two network interfaces. Figure 5.1 illustrates the required blocks for a two-port bridge, with one wired BM and one wireless BM. In order to support the required functionalities, there must be a set of mechanisms related to the IDP and to the Inter-Domain Mobility Procedure (IDMP). Consequently, it is necessary to modify the PROFIBUS MAC in order support the IDP IDTs handling capabilities and the operation of the IDMP. A BM must also contain the routing and the IDF handling functionalities which are crucial for the IDP. These two functionalities are associated with two data structures: the *Routing Table* (RT) and the *List of Open Transactions* (LOT). The Global Mobility Manager (GMM), Domain Mobility Manager (DMM) and the Bridge Master (BM) functionalities are related to the IDMP protocol. From this set, only the BM functionality is mandatory. The assignment of the DMM and GMM roles is done according to the IDMP requirements.

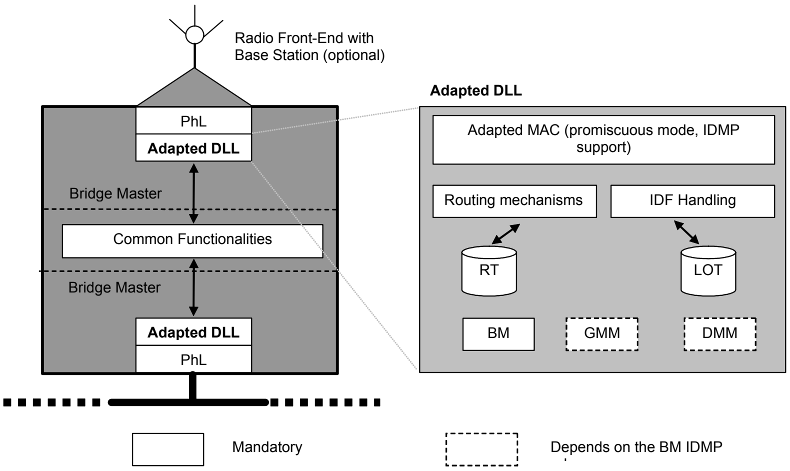


Figure 5.1 – Bridge components

Figure 5.1 also depicts the *Common Functionalities* box, which supported by a shared memory area is responsible for the communication between the two BMs in the bridge.

5.2.1. Adapted MAC

In PROFIBUS, when an initiator makes an Inter-Domain Request, all stations belonging to the initiator's domain discard the frame. This is a standard behaviour for such type of network protocol, since the `Destination Address (DA)` does not match any of the MAC addresses in that domain. However, the IDP requires that the BMs must accept all frames and process it. Therefore, this functionality requires the adaptation of the PROFIBUS MAC sub-layer in the BMs in a way that they process all the frames regardless of the `Destination Address` specified in the frames. Additionally, both the BM_{ini} and the BM_{res} MAC must be capable of transmitting frames containing `Source Address` fields with a value different from their DLL addresses, when the BM_{ini} is replying to the initiator with a stored reply and when the BM_{res} is transmitting the request to the IDT responder.

The IDMP protocol also requires some special operating modes and new frame types which must also be supported by the adapted MAC layer. The new operating modes involve changes on the legacy MAC layer state machine, as will be detailed later in Chapter 6.

5.2.2. Routing Tables (RT)

The IDP approach imposes the use of a single address space, where every station in the overall network has a unique MAC address. Therefore, the routing mechanism relies on the information contained in the routing table to determine if a frame should be relayed to the other BM or not. Therefore, when a BM receives a frame, it verifies its routing table entry related to the addressed station. If the frame is to be forward to the other BM of the bridge, then a "Y" symbol is associated with the address. Otherwise a "N" symbol exists. The routing table also contains another field, the station type, which is only used by the IDMP-related functionalities, as described later in Chapter 6.

Considering the network example depicted in Figure 4.9, Table 5.1 illustrates the routing tables associated to bridge B2. The routing table specifies if a frame addressed to each of the network stations and arriving to BM M7 or M6 should be relayed to another BM (Y) or not (N).

It is assumed that the tables are configured in the bridge prior to run-time. At run-time, routing tables can be dynamically updated, to reflect the inter-domain mobility of stations. As it will be described in Chapter 6, this is achieved through the use of `Route_Update` messages that specify the (new) location of the mobile stations. The joining/leaving of stations in a domain is managed by the ring maintenance mechanisms defined in the PROFIBUS protocol, associated with the `List of Active Stations` (LAS) and the `Live List` (LL) tables.

Table 5.1 – Routing Table example for B2 of Figure 4.9

Destination	**M6**	**M7**
M1	N	Y
M2	Y	N
M3	N	Y
M4	Y	N
M5	Y	N
M6	N	Y
M7	Y	N
M8	Y	N
M9	Y	N
S1	N	Y
S2	N	Y
S3	N	Y
S4	Y	N
S5	Y	N
S6	Y	N
S7	Y	N

5.2.3. List of Open Transactions (LOT)

A BM belonging to the domain of an initiator and acting as a BM_{ini} for certain message transactions must be capable of matching a response to the related pending request. This is achieved using the information contained in the IDF embedding the response, and by using the *List of Open Transactions* (LOT) as described next.

The LOT contains the following information about the request frame: `Destination Address` (DA), `Source Address` (SA), `Destination Address Extension` (DAE) and `Source Address Extension` (SAE). It also contains a transaction identification tag and the state of an open transaction. The `Transaction Identifier` (TI) must be included in both the Inter-Domain request frame (IDreq) and in the associated Inter-Domain response frame (IDres).

Figure 5.2 depicts a state machine which controls the evolution of a LOT entry.

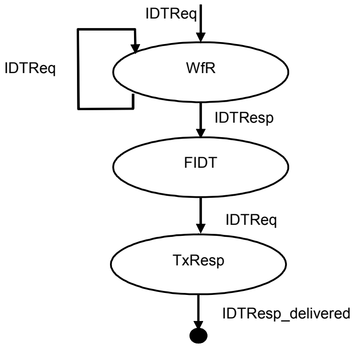


Figure 5.2 – IDT state machine at BM_{ini}

An open transaction is initialised when an IDT request is received by the BM_{ini} (transition `IDTReq`) and the state machine evolves to `Wait_for_Response` (`WfR`) state, where the BM_{ini} waits until the reception of the corresponding response. The response is retrieved from the responder station using the IDP. When a response related to an open IDT arrives at the BM_{ini} (transition `IDTResp`), the LOT entry evolves into the `Finished_IDTs` (`FIDT`) state, until a new request, related to the same transaction is received (transition `IDTReq`) and the state machine evolves to the `Transmit_Response` (`TxResp`) state. Then, when the stored response is transmitted to the initiator (transition `IDTResp` delivered), the records regarding the open transaction are cleared from the LOT, and the state machine ends.

The LOT is also used to handle repetitions of the same request. Thus, for every arriving request, a BM_{ini} consults its LOT, and if that request is already listed, then it is discarded. Table 5.2 shows the content of the LOT just after starting an IDT between M3 and S6, according to the network example depicted in Figure 4.10, when the LOT state evolves to `WfR`.

Table 5.2 – Content of the LOT (example)

DA	SA	SAE	DAE	TI	State
S6	M3	40	50	22	WfR
-	-	-	-	-	-

When an IDF embedding a response arrives at the BM_{ini}, the associated request is searched in the LOT, and the response is associated and stored. This response is returned to the initiator when it repeats the original request. Section 5.2.4 details the format of the IDFs.

5.2.4. Frame Formats

Inter-Domain Frames (IDF) are used by the IDP for proper transmission of frames between bridges. When these frames are exchanged between bridges, they must contain information that enables decoding the embedded original request/response and matching the information stored in the BM_{ini} LOT and the respective response.

The PROFIBUS protocol allows a request using a Variable Data Field Length frame with a `Destination Address Extension` (DAE), to be answered by a Fixed Length Response frame without data field (thus not supporting DAE). So, the BM_{ini} would not be capable of matching two different transactions from the same initiator, addressed to the same responder, but with different DAE. The PROFIBUS DLL protocol also defines that requests using Variable Data Field Length frames can be replied with a `Short Acknowledge` (SC) frame. Obviously, if no special IDF format was used, the bridges would be unable to route the SC frame back to the initiator station, since that type of frame does not have a `Destination Address` (DA) field.

The first problem can be solved by using a `Transaction Identifier`, which enables matching the request and the respective response, while to solve the second problem it is required that every IDF must have a `Destination Address` field.

The TI is a sequence number, assigned by the BM_{ini}, which must also be included in the response frame (similar to a TCP/IP sequence number). This field is used by the BM_{ini} to distinguish between response frames related to different pending transactions.

The IDF is a (new) PROFIBUS frame that embeds the original PROFIBUS frame. Therefore, to reconstruct the original frame, one of the fields that must be stored in the IDF frame is the original frame function code (which is stored as Embedded frame Function Code (EFC)) and an identifier which enables BMs BM_{ini} and BM_{res} to identify the type of the embedded frame – the Embedded Frame Type (EFT).

Considering the three types of data frames defined in PROFIBUS, the IDP converts "Frames of Fixed Length with no Data Field" to "Frames of Fixed Length with Data Field", and both "Frames of Fixed Length with Data Field" and "Frames with Variable Length Data Field" to "Frames with Variable Length Data Field". Table 5.3 summarises the proposed mappings between standard PROFIBUS frames and IDFs. In the table, a rectangle with a dash means that the field is not used in the IDF because it is not present in the original frame. A rectangle with diagonal stripes means that the field is not available to the IDF (in this specific case, "Fixed Length Frames with no Data Field" are mapped into frames of "Fixed Length Frames with Data Field"). The symbol "=" means that the field must be equal to the original embedded frame field.

In the conversion, the IDFs preserve the same DA and SA, except in the case of the Short Acknowledge frame, which does not have DA or SA. In this case, the IDF includes the DA and SA obtained from the request frame.

Table 5.3 – Mapping between standard PROFIBUS frames and IDFs

Original Type of Frame		**Frame Header (PROFIBUS defined)**					**Frame Data (IDP defined)**					
		LE	**SD**	**DA**	**SA**	**FC**	**DAE**	**SAE**	***TI***	***EFT***	***EFC***	**Data Unit**
Fixed length no data	Req	▨	SD3	=	=	10	-	-	TI	1	EFC	-
	Ack	▨	SD3	=	=	10	-	-	TI	2	EFC	-
	Short ack	▨	SD3	Req. SA	Req. DA	10	-	-	TI	3	EFC	-
Fixed length w/ data	Req	Data len	SD2	=	=	10	=	=	TI	4	EFC	=
	Res	Data len	SD2	=	=	10	=	=	TI	5	EFC	=
Var. length	Req	Data len	SD2	=	=	10	=	=	TI	6	EFC	=
	Res	Data len	SD2	=	=	10	=	=	TI	7	EFC	=

To distinguish IDFs from other frame types, the Function Code of the FC field must be equal to 10 (note that this feature also imposes a non standard behaviour by the BMs DLL). Its remaining sub-fields should be filled with the appropriate values (for a PROFIBUS frame). All frames defined in Table 5.3 are transmitted as individual requests. Also note that the response frame (transmitted by the responder) is coded and transmitted as a request frame by the BM_{res}. Finally, SDN frames do not need any conversion, therefore they can be relayed by the bridges as received (without being coded).

In this approach, the maximum size of the data unit is reduced by 3 bytes, i.e. to 241 bytes in frames using address extension, and to 243 bytes in frames without address extension. Nevertheless, this overhead of the protocol has a minor impact on network performance, since in most PROFIBUS applications frames are typically short.

5.2.5. Error Control

PROFIBUS already includes some mechanisms for the detection of errors in an IADT (like the `Slot Time` timer) and the capability of making retries. Nevertheless, none of these mechanisms can be adapted to insure the proper handling of errors in a Multiple Logical Ring (MLR) PROFIBUS network. Therefore, a new mechanism to handle message loss/errors in IDTs is proposed next.

The proposed mechanism is controlled by the BM_{ini}, which starts an *IDT Error Handling Timer* for every open transaction in its LOT. If the IDT is still open at the expiration of the timer, then the open transaction is cleaned from the LOT.

The *IDT Error Handling Timer* must be set to a value larger than the worst-case time required by bridge BM_{ini} to complete any of its IDTs – $T_{EH} = max(Rbmi_i^k)$ (detailed in Chapter 7). Note that the BM is not capable of identifying the specific timing parameters of every IDT, which use the BM as its BM_{ini}. Consequently, the value of T_{EH} is equal for every transaction. Assuming the network configuration illustrated in Figure 4.9, Figure 5.3 depicts an example regarding an IDT (S_1^{M3}) between M3 and S6.

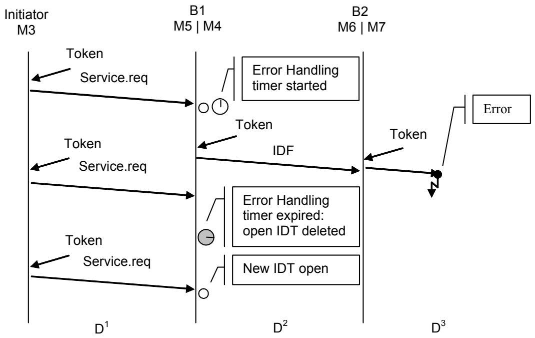


Figure 5.3 – IDTs error scenario illustration

In this example, the *IDT Error Handling Timer* is loaded with T_{EH}, when the first service request related to transaction S_1^{M3} is received. The IDreq is relayed by the system bridges, but the transaction with S6 is not completed and bridge B2 does not detect the problem (e.g. due to an hardware error). Since the transaction S_1^{M3} is not completed, at the expiration of the timer, this transaction is deleted from the LOT. Therefore, the next request (of the same type) will open a new IDT on the BM LOT.

There is also the possibility that one of the BMs on the IDT communication path detects the error. In that case, the BMs can simply transmit an `Error Signalling` message, addressed to the initiator, specifying that the transaction cannot be completed. That message only needs to include the `Transaction Identifier` field.

Table 5.4, depicts the structure of a `Error Signalling` message, which is mapped into a PROFIBUS Fixed Length frame with data field. Its main difference in relation to other standard PROFIBUS frames is related to the `Function Code` (FC) field, which, in this case is set to 11 (a reserved value). The `Destination Address` field is equal to the initiator address (the `Destination Address` of the IDF). The `Destination Address` (DA) field is equal to the BM which detected the error and the data unit is empty.

Table 5.4 – Mapping between standard PROFIBUS frames and IDFs

Frame Header				Frame Data	
SD	**DA**	**SA**	**FC**	***TI***	**Data Unit**
SD2	Init	BM	11	TI	-

One consequence of the proposed error handling method is the possibility of IDFs duplication, which happens when there is an abnormal delay on the reception of an IDF containing the response relative to an open IDT by BM_{ini}. Such an abnormal delay may occur when there is a token loss in one of the domains in the IDT communication path. As described next, the proposed error control mechanism is also capable of overcoming the IDF duplication problem.

Figure 5.4 illustrates a scenario of a transaction performed between two stations, belonging to different domains of a network. In this scenario we assume that that an IDreq frame (IDreq1), associated with the transaction, is delayed during its relaying by the bridges. Notwithstanding that fact, the message cycle with the responder is completed, and the IDres frame (IDres1) carrying the response is routed back to BM_{ini}. Meanwhile, the IDT *Error Handling Timer* (at BM_{ini}) expires, and the initiator issues again the same request, opening a new entry in the LOT (with a new `Transaction Identifier`). Now IDreq frame (IDreq2) is relayed through the network.

When IDres1 reaches BM_{ini} it tries to match the TI of the IDF carrying the response, related to the first opened IDT in the LOT, with the TI in its LOT, therefore detecting an error. Consequently, BM_{ini} discards the frame.

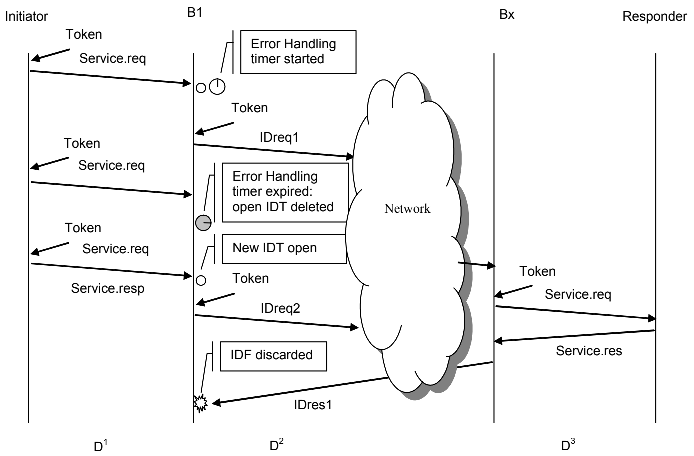


Figure 5.4 – Another IDT error scenario illustration

5.3. Cooping with Compatibility between IDP and PROFIBUS-DP

Although the IDP has been designed with the objective of being compatible with PROFIBUS, there are some issues in which the IDP can have some impact on the operation of PROFIBUS-DP. A concise description follows.

The initialisation of a PROFIBUS slave is executed by masters through a sequence of commands which require immediate responses by the slave, otherwise the process is restarted (EN50170, 1996). In a bridge-based network, if a master and a slave are not in the same domain that kind of operation fails (for details consult Section 2.2.3).

There are at least three possibilities to solve this problem.

1. The master and the slave must be in the same domain during the initialisation phases.
2. One of the bridges, located in the domain to which the slave belongs, configures the slave and returns the slave state to a bridge belonging to the domain where the master is located. In this way the master does not notice that the slave is not in its domain.
3. The BM_{ini} of the IDT between a master and a slave is responsible for controlling the procedure in a transparent way to the master and the slave.

The first solution proposed is the simplest, but it may be impractical if neither the slave nor the master are able to be located in the same domain (e.g. if one is a wired station and the other is a wireless station).

The second case requires the bridges to be loaded with the configuration parameters of the slaves, withdrawing some of the transparency to the proposed bridge-base solution. At the same time the system requires a more complex maintenance, especially in case the slave parameters have to be changed.

The third solution offers a behaviour which is completely transparent from the point of view of the masters and slaves involved. The description of the proposed solution, supported by Figure 5.5 and Figure 5.6, will be detailed next.

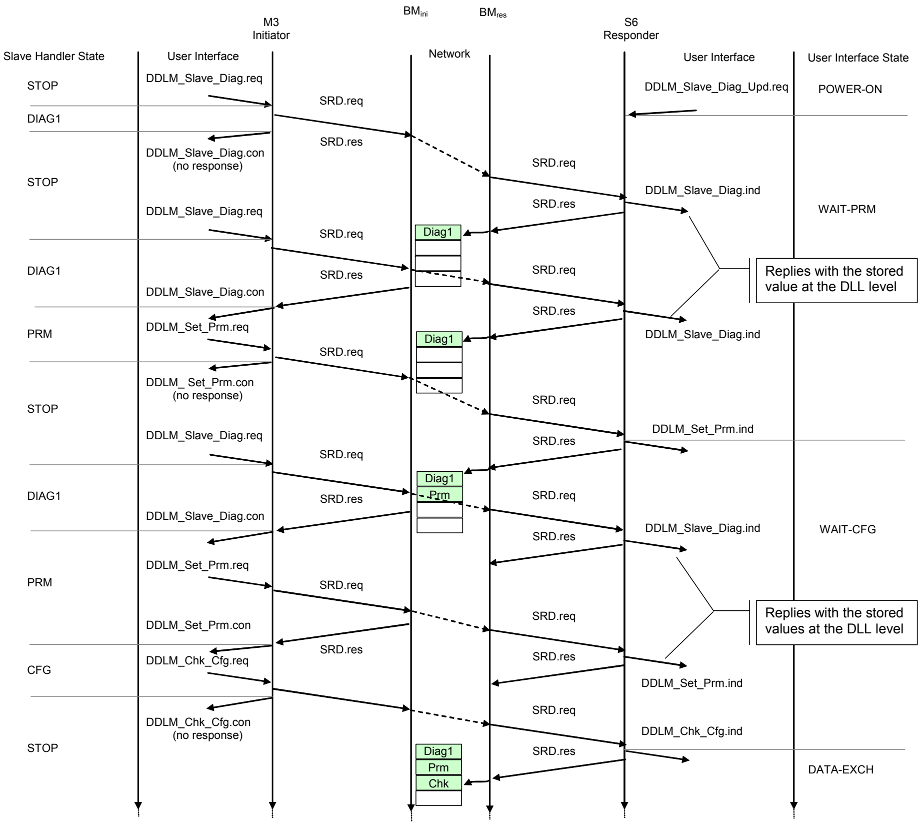


Figure 5.5 – Slave initialisation procedure in a bridge-based network

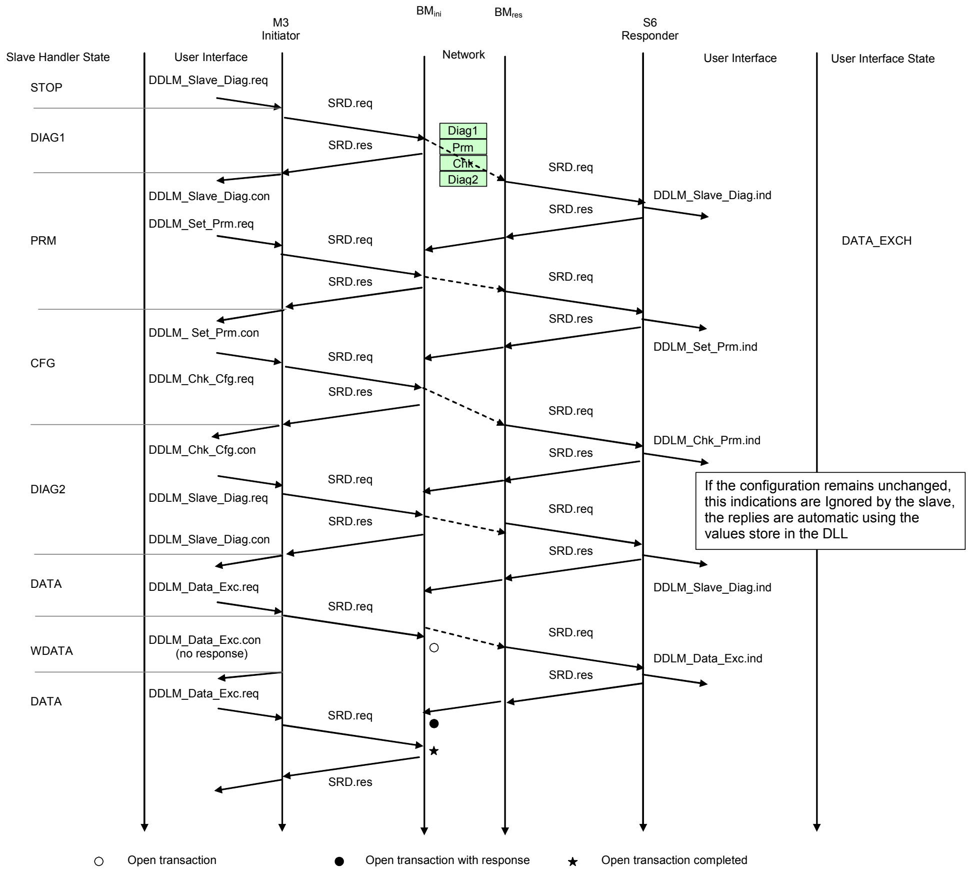
The proposed protocol depends on the recording of the slave's responses, by the IDT BM_{ini}, to the slave's initialisation services: DDLM_Slave_Dig (repeated twice during the initialisation phase), DDLM_Set_Prm and DDLM_Chk_Cfg. This protocol also relies on the services provided by the IDP to communicate between the master and the slave.

The procedure operates as follows:

- when BM_{ini} first detects a DDLM_Slave_Dig.req, it initialises a 4 position buffer, the *Slave Initialisation Data Buffer*, which will be used to store the slave responses to the initialisation services;
- BM_{ini} stores in the *Slave Initialisation Data Buffer* the values regarding the responses from the slave to the initialisation services;

- if the BM_{ini} has a value stored on its *Slave Initialisation Data Buffer*, then it answers to the master using that value, otherwise it will not reply;
- it is expected that the master continuously, tries to initialise the slave (this feature is inherent to the master *slave handler state machine*);
- if the slave is located on the master domain, then the procedure is disabled.

The effect of the described procedure is that BM_{ini} will be able to incrementally update the *Slave Initialisation Data Buffer* with the data collected in the slave.



If the configuration remains unchanged, this indications are Ignored by the slave, the replies are automatic using the values store in the DLL

Figure 5.6 – Slave initialisation procedure in a bridge-based network (cont.)

From the point of view of the initiator it starts by executing the service `DDLM_Slave_Diag`, and its *slave handler state machine* evolves to the DIAG1 state, since no response is received the state machine returns to the STOP state and a confirmation without data is returned to the DP Application Layer (AL). Meanwhile, BM_{ini} receives the `DDLM_Slave_Diag` response and stores it in the *Slave Initialisation Data Buffer*. After, the initiator restarts the procedure, sending a `DDLM_Slave_Diag.req`, to which a response is available at BM_{ini}. Therefore the initiator *slave handler state machine* evolves from the DIAG1 state to the PRM state, and it tries to send the configuration parameters to the slave using the DDLM_Set_Prm service. Since *BM_{ini}* does not have a response to this service a confirmation without data is returned to the DP AL of the initiator, and the *slave handler state machine* returns to the STOP state. Since the initiator keeps trying to configure its slaves, this procedure continues until *BM_{ini}* has response to all of the initialisation services.

From the point of view of the slave, it stores the responses to the initialization services in its DLL using the update services provided by the DP AL, therefore a response to any master request is available prior to the reception of the service frame. If there are no parameter changes, the services DDLM_Set_Prm and DDLM_Chk_Prm do not generate any errors.

Figure 5.6 continues the sequence of events assuming that meanwhile *BM_{ini}* has obtained all responses to the initialisation services, and the *slave state machine* is on the DATA_EXCH state.

In Figure 5.6, since *BM_{ini}* has all responses to the initialization services the *slave handler state machine*, on the initiator, evolves until attaining the DATA state, in which it can freely exchange data with the slave using the service DDLM_Data_Exc. The slave remains in the DATA_EXCH state during the complete procedure.

PROFIBUS-DP also defines the Sync and Freeze Services, which are used by to synchronise the operation of variable reading and writing, as detailed in Section 2.4.3. Such types of services assume that the frames containing the Sync and Freeze commands are received by all stations simultaneously. Obviously, that situation is impossible on a bridge-based network, since the stations in the network receive those commands at different times due to the latencies caused by the relaying process. Therefore, most of the functionality provided by the Sync and Freeze Services is lost.

5.4. IDP Implementation Approach

We assume that when a BM receives a request, it calls the *Indication_Handler* function. This function is responsible for forwarding the original request coded using the IDP to the other BM, and for replying to the initiator station when the response frame is available (at the *BM_{ini}*). It is also assumed that the reception of a response by the *BM_{res}* will be handled by a *Confirmation_Handler* function, which codes the response using the IDP and forwards it to the other BM. Both these functions are detailed next.

*5.4.1. **Indication_Handler** Function*

The Indication_Handler function (Figure 5.7) starts by checking the received frame (variable req_frame), in order to determine the operation that will follow (line 5). For that propose, it uses the req_frame Destination Address, Source Address and FC code together with the information contained in the BM (RT, LL and LAS).

If the req_frame is addressed to a BM, then it is processed by its protocol stack and replied (lines 9-12).

If the bridge receives an IDF, then this frame is forwarded to the other BM, using the `Fwrd_ID_Request` function (described later in Figure 5.8). In the case that the initiator station belongs to the domain of the BM and the `req_frame` is addressed to a station in another domain, then the BM must try to initialise the LOT with another pending IDT using `Init_ID_Request` function (described later in Figure 5.9)**.**

Broadcast frames must also be relayed to other domains and, at the same time, be processed by the bridge. In the other cases (e.g. when the frame is addressed to a station belonging to the same domain as the BM), the BM will not process the `req_frame` (lines 26-29).

```
1.  Indication_Handler(req_frame)
2.  {
3.  // Checks the req_frame to determine the operation to follow
4.   res = check_addr(req_frame)
5.
6.   Switch (res)
7.   {
8.  // When the req_frame is addressed to the bridge
9.     case ST_ADDRESS:
10. // Process the message according to its contents
11.       process(req_frame);
12.    end;
13.
14. // When the req_frame is an IDF that must be forwarded by
15. // the BM
16.    case FWRD_ID_REQUEST:
17.       Fwrd_ID_Request(req_frame);
18.    end;
19.
20. // When the initiator is on the BM domain
21.    case IDT:
22.       Init_ID_Request(req_frame);
23.    end;
24.
25. // req_frame sent in broadcast
26.    case BROADCAST:
27.      process(req_frame);
28.      Fwrd_ID_Request(req_frame);
29.    end;
30.
31.    default:
32.    end; // Do nothing
33.  }
34. }
```

Figure 5.7 – `Indication_Handler` function, pseudo-code algorithm

The `Fwrd_ID_Request` function (Figure 5.8) is called by the `Indication_Handler` function when it receives an IDF and it operates with the resources of the other BM (of the bridge).

The `Fwrd_ID_Request` starts by determining if the destination station is on its domain (line 3). If not, then the frame is queued to the output queue of the BM (line 32). Otherwise, the function determines the type of Inter-Domain Request (line 7). This function is also called by the Confirmation_Handler function after BM_{res} has received a response to its request.

If the ID_req_frame embeds a response frame that matches one entry in the LOT, then the ID_req_frame is decoded and a response (using the standard PROFIBUS format) is stored (lines 10 to 18). If the ID_req_frame embeds a request frame, then this frame is decoded. The information concerning this request is stored in order to enable the identification of the related response. Additionally, the frame is put in the output queue (using the standard PROFIBUS format) (lines 22 to 28).

```
1.  Fwrd_ID_Request(ID_req_frame)
2.  {
3.   res = is_station_on_domain(ID_req_frame);
4.   if res == 1 then // on the domain
5.   {
6.     // Det. type of frame
7.     type = type_of_ID_req(ID_req_frame);
8.     if type == RESP then
9.     {
10.       // Accesses the LOT to find match
11.       res = LOT_match_resp(ID_req_frame);
12.       // if yes, stores the corresponding reply
13.       if res == 1 then
14.       {
15.          // Decodes the ID_req_frame
16.          std_resp_frame = prepare_std_resp(ID_req_frame);
17.          store_reply(std_resp_frame);
18.       }
19.    }
20. else // type = Request
21.    {
22.       // Decodes the request embedded in the ID_req_frame
23.       std_req_frame = prepare_std_req(ID_req_frame);
24.       // Stores the information necessary to identify the
25.       // respective confirmation
26.       Store_info(std_req_frame);
27.       queue(std_req_frame);
28.    }
29.  }
30.  else // station in another domain
31.  {
32.    queue(ID_req_frame);
33.  }
34. }
```

Figure 5.8 – Fwrd_ID_Request function, pseudo-code algorithm

The Init_ID_Request function (Figure 5.9) is also called by the Indication_Handler function when it needs to initialise a pending IDT in the LOT. The Init_ID_Request function starts by checking (in the LOT) if there is another entry with the same data (line 4). In the affirmative case, the bridge will not do any additional processing on this frame. Otherwise, it stores data relative to this pending IDT in the LOT, and starts a count down timer.

```
Init_ID_Request(req_frame)
{
 // Test if there is a match with any
 // other pending transaction on the LOT
 res = check_LOT(req_frame);

 if res != 1 then // No entry on the LOT
 {
 // Updates the LOT
   handler = updt_LOT(req_frame);
   start_error_handling_timer(handler);

 // Codes an IDF
   ID_req_frame = prepare_IDF(req_frame)
   Fwrd_ID_Request(ID_req_frame);
 }
 else // There is a match on the LOT
 {
 // Do nothing
 }
}
```

Figure 5.9 – `Init_ID_Request` function, pseudo-code algorithm

5.4.2. `Confirmation_Handler` Function

The `Confirmation_Handler` function is called when a bridge receives a response to a request. In Figure 5.10, we are only detailing the part relative to the response to an Inter-Domain Request.

This function starts by determining the message type, using the `Destination Address` of the `resp_frame` and the information stored by the `Forward_ID_Request` function.

If the received frame is a response to an Inter-Domain Request, then the bridge prepares a new frame (an IDF) using the IDP and forwards it through the other BM. The other cases handle standard PROFIBUS functionalities, e.g. any request addressed to the bridge.

```
Confirmation_Handler(resp_frame)
{
 res = type_of(resp_frame)
 Switch (res)
 {
   case INTER_DOMAIN_RESP:
    ID_req_frame = prepare_ID_req(res_frame, req_data1);
    Fwrd_ID_Request(ID_req_frame);
   end;
…
 }
}
```

Figure 5.10 – `Confirmation_Handler` function, pseudo-code algorithm

5.5. Example Scenario

In order to improve the intuition on the IDP, in this section we describe an example of one Inter-Domain Transaction (IDT), considering the network scenario depicted in Figure 5.11. It is assumed that the traffic in the network is restricted to the token passing and to one IDT between master M3 and slave S6, in domain D^1 and domain D^3, respectively.

In this example, we denote a message frame as *Station_ID.n*, where *Station_ID* is the station or the BM identifier (e.g. M3) which transmitted the frame, and *n* is a (sequential) number identifying that message.

According to the IDP, request M3.1 must be repeated several times until the actual response from S6 is received. In this example we are assuming no retries at the DLL level. Nevertheless, 3 repetitions of the same request are executed at the AL layer level. In Figure 5.11, the temporal order of the frames is indicated by the numbers inside the circles next to the message notations. Figure 5.12 depicts the actual timings for these frames.

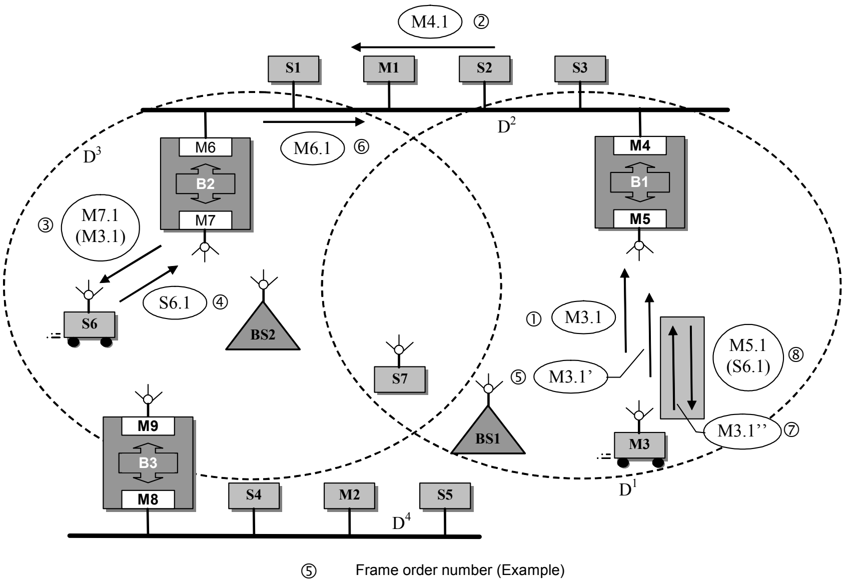


Figure 5.11 – Inter-Domain Protocol illustration

The first request issued by M3 (M3.1) is addressed to S6, thus BM M5 converts the frame using the IDP and relays it to D^2, without sending any reply to M3. Since M3 belongs to the same domain as BM M5, the latter adds a pending IDT to its LOT.

Frame M4.1, transmitted by M4, preserves the Destination and Source Addresses of the original request. So, BM M6 receives the frame and forwards it to BM M7.

Since BM M7 knows that S6 belongs to the same domain, then M7 must decode the original request frame (M3.1), embedded in frame M4.1, and transmit it to D^3. Then, S6 receives frame M7.1 and responds. Note that frame M7.1 is equal to frame M3.1.

After receiving the response from S6 (S6.1), BM M7 converts it using the IDP and relays it to D^2, with `Destination Address` M3 (M6.1). Meanwhile, M3 repeats request M3.1 (M3.1'). When BM M5 receives that repeated request, it consults its LOT and detects that a similar transaction is already going on, so it takes no action (discarding the repeated request).

The response to request M3.1 is received by bridge B1 embedded in the IDF frame M6.1. Since the destination station belongs to the same domain as M5 and there is a related entry in the LOT, then M5 stores the response to M3.1 in a format equal to response S6.1.

When M3 issues another repetition of request M3.1 (M3.1''), BM M5 replies using frame M5.1 (equal to S6.1) and closes that pending transaction (the corresponding entry in the LOT is deleted).

Figure 5.12 shows a timeline related to the timings of this IDT.

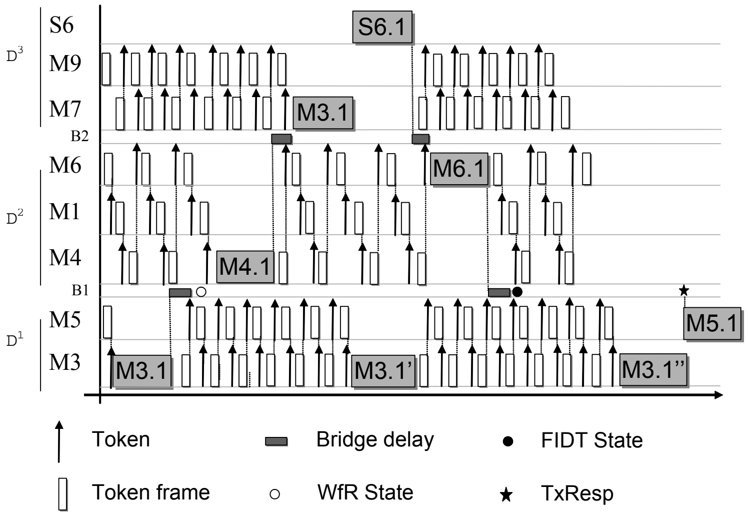


Figure 5.12 – Timeline example for an Inter-Domain Transaction

In this figure it is noticeable the influence of the different (independent) token rotations on the overall latencies of this particular transaction and the delay that exists in the bridges before a request is decoded and converted using the IDP (in this particular case the delays represented in the figure are exaggerated; in a real situation they are supposed to be much smaller). In this example, when bridge B1 receives the first request (M3.1), BM M5 initialises a pending transaction in its LOT (the LOT entry state evolves to `WfR` – Wait for Response). This transaction will only be deleted from the LOT when the respective response is received (when the LOT entry state evolves to `FIDT` – Finished IDT), and after request M3.1'' has been transmitted to M5 (when the LOT entry state evolves to TxResp – Transmit Response). For further details on the state machine that controls the LOT, the reader is referred to Section 5.2.3.

Details on the exchanged messages are provided in Table 5.5, considering the following assumptions. M3 and S6 have DLL addresses of 3 and 38, respectively, and M3 is using Service Access Point (SAP) 3 in its Source Address Extension field and SAP 4 in its Destination Address Extension field. In the case of request frames M3.1 and M7.1, FC is equal to 13, which codes a standard Send and Request Data service, using a PROFIBUS Variable Data Field Length frame. Frames M4.1 and M6.1 are IDFs exchanged between bridges. Therefore, they are coded according to the IDP, using the reserved FC value of 10. Frames S6.1 and M5.1 are response frames. For that reason FC is equal to 10, in the case of frame S6.1 this value is set according to the PROFIBUS standard and, in the case of M5.1 the frame is transmitted as a request coded according to the IDP. Finally, the length of the data field of the request and response is equal to 11 and 16 bytes, respectively. The values for the fields LE, DAE, SAE and TI had been arbitrated.

Table 5.5 – Details on the exchanged messages

Frame	Frame Header					Frame Data					
	LE	SD	DA	SA	FC	DAE	SAE	*TI*	*EFT*	*EFC*	Data Unit
M3.1	11	SD2	38	3	13	4	3	▨	▨	▨	…
M4.1	14	SD2	38	3	10	4	3	167	6	13	…
M7.1	11	SD2	38	3	13	4	3	▨	▨	▨	…
S6.1	16	SD2	3	38	10	3	4	▨	▨	▨	…
M6.1	19	SD2	3	38	10	3	4	167	7	10	…
M5.1	16	SD2	3	38	10	3	4	▨	▨	▨	…

5.6. Summary

This chapter addressed how the IDP can be supported by bridge devices. The IDP enables the execution of transactions between standard PROFIBUS stations belonging to different domains of a hybrid wired/wireless PROFIBUS bridge-based network. The chapter described in detail the architecture for the bridges, focusing on the routing mechanisms, handling of IDT transactions, frame formats and error control. In addition, this chapter also tackled some issues regarding the compatibility between the IDP and PROFIBUS-DP and included a detailed description of the IDP implementation in the bridges.

The following chapter specifies the Inter-Domain Mobility Procedure, which enables mobile wireless stations and mobile wired domains to move transparently between different wireless domains.

Chapter 6

The Inter-Domain Mobility Procedure (IDMP)

This chapter addresses the specification of the Inter-Domain Mobility Procedure (IDMP), which enables a mobile wireless station or a mobile wired domain to move between different wireless domains. The proposed mechanism guarantees transparent and reliable transactions, no loss, and orderly delivery of messages. Moreover, the IDMP is compatible with the PROFIBUS protocol in the sense that only Bridge Masters (BM), mobile wireless stations and mobile linking intermediate system must implement the IDMP functionalities.

6.1. Introduction

As already outlined in Section 2.3, in RFieldbus (Alves *et al.*, 2002) the mobile wireless stations perform channel assessment and (eventually) channel switching in a periodic fashion. The mobility management mechanism is triggered by one of the masters (the mobility master), which periodically triggers a radio channel assessment and switching phase. Each Structuring Intermediate System (SIS) sends `Beacons` on its own radio channel. These `Beacons` are special frames with a specific format (therefore not standard PROFIBUS frames), which are then used by the mobile wireless stations to assess the quality of the radio channels transmitted by the different SISs. At the end, the wireless stations switch to the channel offering the best signal quality. Note that, as there is only one token rotating (single logical ring system and broadcast network), there is no message loss and no need for specific registering mechanisms in the SISs.

The use of bridge-like Intermediate Systems (IS) requires a more elaborated mobility management procedure, since we are dealing with a Multiple Logical Ring (MLR) system. Therefore, mobile wireless stations must be provided with similar mechanisms to assess radio channel quality, but further mechanisms to support stations (not only the masters) leaving and joining a logical ring (domain) must be provided as well.

In (Ferreira, *et al.,* 2002), the authors described the possibility of using the native PROFIBUS ring management mechanisms to support inter-domain mobility. The mobility of master stations can be tackled by the PROFIBUS GAP Update mechanism. Slave stations leaving a domain can be detected either when the station does not reply to a request or by the use of the PROFIBUS `Live List` (LL) mechanism. A slave joining a logical ring can also be supported by the LL mechanism or when it replies to a request addressed to it. However, stations joining and leaving a wireless domain trigger the need for updating routing tables in the bridging devices. This is mandatory for the Inter-Domain Protocol (IDP) to work properly, when handling Inter-Domain Transactions (IDT).

These are just the basic ideas of the proposed IDMP. Additional mechanisms must however be added to guarantee no errors, no loss, and orderly delivery of frames concerning Inter-Domain Transactions (IDT).

The following section (Section 6.2) details the IDMP which overturns these problems, and Section 6.3 presents an example scenario which illustrates the operation of the IDMP. Later, we go into deeper details about the implementation of the IDMP (Section 6.4) and on the format of the mobility-related messages (Section 6.5).

6.2. Description of the IDMP

6.2.1. Problems to be Addressed by the IMDP

One of the main problems of any mobility management procedure is to guarantee the orderly delivery of frames in transactions that involve mobile wireless stations, since the temporal sequence of messages must be assured.

PROFIBUS has been designed based on the presupposition of a “broadcast” communication network, in which all messages are received by all stations. While it offers some limited functionalities for guaranteeing the detection of two consecutive missing frames, these are not adequate when considering a network architecture with multiple logical rings supporting inter-domain mobility, such as the one proposed in this thesis.

These functionalities are usually available on specific protocols designed for multihop networks, like TCP/IP. In TCP/IP, every frame includes a *Sequence Number* field that is used by the protocol to correctly reassemble several fragments into a single frame. This field is also used by the destination station to detected missing frames. In such type of networks, the protocol also provides the necessary features in order to request the retransmission of the missing frames.

To illustrate the problems related to mobility in a bridge-based PROFIBUS network, the scenario depicted in Figure 6.1 is assumed. In this example, the temporal order of the messages is the following: {M3.1, M4.1, M3.2, M7.1}. During the time that elapses from transmission of messages M3.1 and M3.2, M3 moves between domain D^1 and D^4. The consequence is that S6 may receive M3.2 prior to M3.1, contrarily to what should happen.

Suppose that M3 is responsible for controlling the movement of S6 (an AGV, for instance). M3 starts by sending a stop command to S6 (M3.1). The message is transmitted, bridge B1 opens an IDT in its List of Open Transactions (LOT) and forwards the frame (M4.1) through the network. Meanwhile, M3 moves from domain D^1 to domain D^4 (neglecting for the moment the mobility procedure in use). When M3 is in D^4, it resumes normal operation and sends another request (M3.2), this time instructing S6 to move. Since M3 now belongs to the same domain as S6, it immediately receives this message (M3.2). Meanwhile, frame M7.1, which embeds request M3.1, is transmitted to S6, carrying the first instruction to stop the vehicle. Figure 6.2 depicts a timeline for this erroneous scenario.

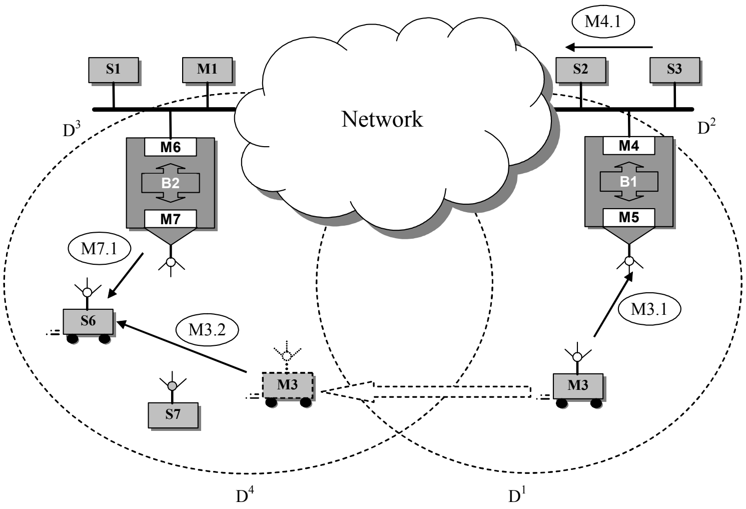


Figure 6.1 – Example scenario

As illustrated by this simple example, the inversion of message order can have a significant impact and must be avoided in any control application.

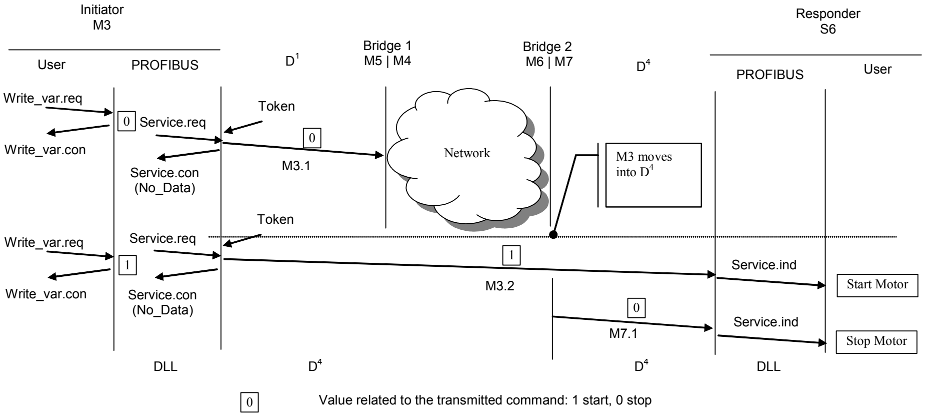


Figure 6.2 – Order inversion problem for inter-domain mobility

Another problem of the bridge-based architecture is related to broadcast frames. In fact, when a broadcast frame is transmitted it may not reach all mobile wireless stations in the network.

The scenario depicted in Figure 6.1, can also be used to illustrate this other problem. Assume that M3 broadcasts a message that is relayed by bridge B1 to all other network domains. Assume also that before this message is able to reach domain D^4, S6 moves to domain D^1. Since the message has already been transmitted in D^1, then S6 will not receive it. This kind of broadcast related problems can only be tackled by specific atomic multicast algorithms (Hadzilacos and Toueg, 1993). Since usually these algorithms are supported at the Data Link Layer (DLL), Network Layer or Application Layer levels, then in order to maintain the compatibility with standard PROFIBUS, this problem is left unsolved by the IDMP.

6.2.2. Agents of the IDMP

The IDMP manages inter-domain mobility in a hierarchical fashion. One master in the overall system implements the global mobility management functionality – the *Global Mobility Manager* (GMM). In each domain, one master controls the mobility of stations belonging temporarily to that domain – the *Domain Mobility Manager* (DMM). Finally, the BMs must implement specific mobility services. Figure 6.3 illustrates, for a particular network scenario, which stations can assume the role of the two previously mentioned mobility management entities (GMM and DMM).

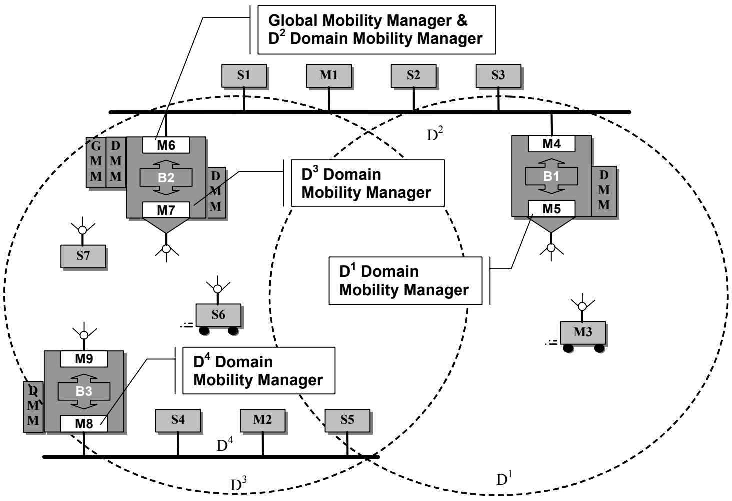


Figure 6.3 – An example network with the location of the IDMP agents

The GMM must know the addresses of all the BMs and DMMs in the system. Each DMM must know the addresses of the BMs in its domain. In the special case of a wireless DMM, it is also required that it knows the addresses of all mobile wireless stations in the network.

The IDMP also assumes that the BMs routing tables include information which identifies the mobile wireless stations and that the BMs are able to modify those entries, according to the information received on their (new) location.

The functionalities of the DMM and GMM might be attributed to any non-mobile wireless master station in the network, but that would require changes to its protocol stack, which would not satisfy our compatibility requirements. Therefore, in the remaining of this thesis, it is assumed that only the BMs may support such functionalities.

In Figure 6.3, station M6 assumes both the role of GMM and DMM of domain D^2. BMs M5, M7 and M8 assume the role of DMMs for domain D^1, domain D^3 and domain D^4, respectively. The mobile wireless stations operate as standard PROFIBUS stations with a wireless interface and must be capable of assessing the quality of the radio channels in the system, like in RFieldbus.

6.2.3. Phases of the IDMP

As already outlined in Section 4.4.2, the IDMP evolves through 4 phases, as shown in Figure 6.4. The objective of these phases is to insure that the procedure will not generate errors, that the inaccessibility periods are minimal (especially in the case of IADTs) and that the mobile wireless stations are able to evaluate all wireless radio channels and switch to the best one seamlessly. The proposed mechanism is synchronous in some of its phases: Phase 1 and Phase 2. But in the case of Phase 3, the ending of it in the domains is not synchronised, and Phase 4 runs asynchrounously for each domain.

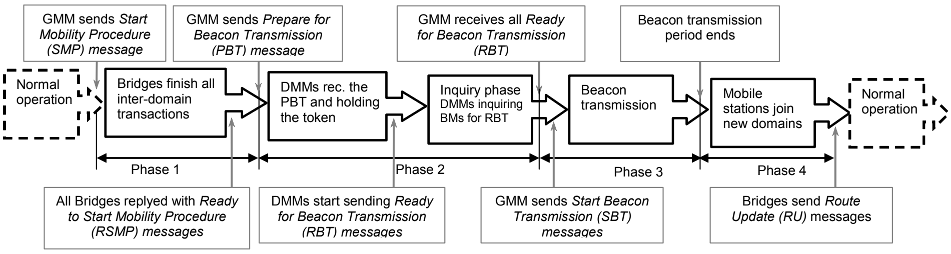

GMM sends *Start Mobility Procedure (SMP) message*
GMM sends *Prepare for Beacon Transmission (PBT) message*
GMM receives all *Ready for Beacon Transmission (RBT)*
Beacon transmission period ends
Normal operation
Bridges finish all inter-domain transactions
DMMs rec. the PBT and holding the token
Inquiry phase DMMs inquiring BMs for RBT
Beacon transmission
Mobile stations join new domains
Normal operation
Phase 1
Phase 2
Phase 3
Phase 4
All Bridges replyed with *Ready to Start Mobility Procedure (RSMP) messages*
DMMs start sending *Ready for Beacon Transmission (RBT) messages*
GMM sends *Start Beacon Transmission (SBT) messages*
Bridges send *Route Update (RU)* messages

Figure 6.4 – IDMP Phases

Phase 1

Phase 1 starts with a `Start_Mobility_Procedure` (SMP) message sent by the GMM. This message is sent periodically according to the mobility requirements (e.g. linear velocity) of the mobile wireless stations involved in the application. All bridges in the system relay the SMP, which triggers a sequence of actions which are outlined in Figure 6.5 (assuming the network scenario depicted in Figure 6.3).

When the BMs receive the `Start_Mobility_Procedure` message, they stop processing new IDTs from the masters belonging to their domains. Nonetheless, they keep handling pending IDTs (still present in their LOTs) and, importantly, they keep relaying IDF originated in other domains. After completing all pending IDTs (those from their LOT), the BMs transmit a Ready_to_Start_Mobility_Procedure (RSMP) message to the GMM. When the GMM has received all Ready_to_Start_Mobility_Procedure messages it starts Phase 2 of the IDMP.

An alternative approach would be to instruct all BMs to delete all ongoing IDTs from its LOTs and output queues, upon reception of the Start_Mobility_Procedure message. The problem with such an approach is that some types of applications involve the sequential and ordered transmission of a set of data by the initiator or by the responder (e.g. in the transmission of stored data recorded by a slave). If ongoing IDTs are deleted, then one or more pieces of data would be lost.

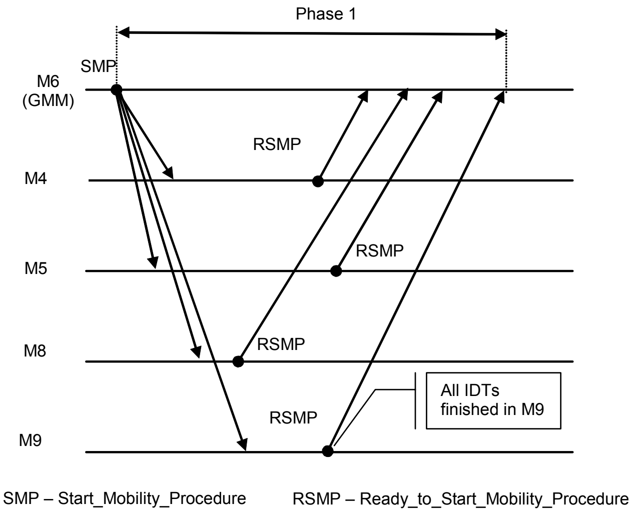

Phase 1
SMP
M6
(GMM)
M4
M5
M8
M9
RSMP
RSMP
RSMP
RSMP
All IDTs finished in M9
SMP – Start_Mobility_Procedure
RSMP – Ready_to_Start_Mobility_Procedure

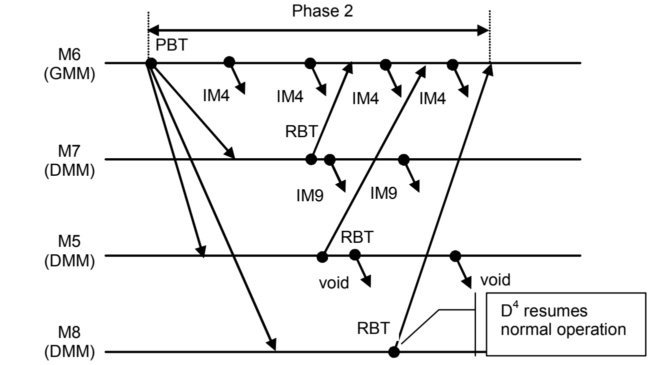
Figure 6.5 – Exchanged messages in Phase 1

Phase 2

Phase 2 is triggered by the GMM broadcasting the Prepare_for_Beacon_Transmission (PBT) message (Figure 6.6). After receiving the PBT message, a DMM holds the token (after token reception, obviously), starting an inquiry sub-phase. When receiving the PBT message all BMs in the network clear their routing table entries related to mobile wireless stations.

On the inquiry sub-phase, the DMMs start by transmitting a Ready_for_Beacon_Transmission (RBT) message to the GMM signalling that they are on the inquiry sub-phase, ready for Beacon transmission. After, every DMM sequentially sends Inquiry frames addressed to the BMs belonging to its domain. The BMs use the response message to transmit any mobility-related message that they require to transmit.

Wired terminating domains (i.e. wired domains connecting to only one bridge) may resume normal network operation (as in the case of Domain D^4, Figure 6.6), consequently in such domains the time during which IADT are disabled is equal to zero. Wireless terminating domains (i.e. wireless domains connecting to only one bridge) emit Void frames (frames without information, define later) in order to maintain network activity. Note that in domains, the DMM does not have to retrieve any mobility-related message from the other bridges in its domain.



PBT – Prepare_for_Beacon_Transmission RBT – Ready_for_Beacon_Transmission

IMx – Inquiry message to BMx

Figure 6.6 – Exchanged messages in Phase 2

This procedure allows a fast communication between the GMM and the DMMs, while at the same time the inaccessibility period of the wired stations is kept small, as it will be shown in Chapter 7.

Phase 2 ends when all Ready_for_Beacon_Transmission (RBT) messages are received by the GMM.

Phase 3

After collecting all Ready_for_Beacon_Transmission messages from all the DMMs, the GMM starts the Beacon transmission sub-phase by broadcasting the Start_Beacon_Transmission (SBT) message (Figure 6.7). Upon reception of this message, Structuring Intermediate Systems start emitting Beacons (similarly to the implementation in RFieldbus).

The mobile wireless stations use the Beacon frames to evaluate the quality of the different radio channels and to decide if they want handoff (or not). So, before the end of the Beacon transmission, every mobile wireless station that wants to handoff must switch to the new radio channel. Figure 6.7 depicts an example stressing the fact that the duration of Phase 3 is different for the different domains, since the beginning of the different Beacon transmission sub-phases is not synchronized, and the duration of the Beacon transmission also differs between domains.

Note that all wired domains that evolved to the inquiry sub-phase may resume IADTs, after the correspondent DMM has received the SBT message, since their intervention is not required on the remaining phases of the IDMP. Note also that, IDTs can be relayed if the neighbouring domains already have their IDTs enabled. IDTs involving mobile wireless stations are only resumed when the BMs belonging to the initiator domain receive `Route_Update` (RU) messages specifying the location of the mobile wireless stations.

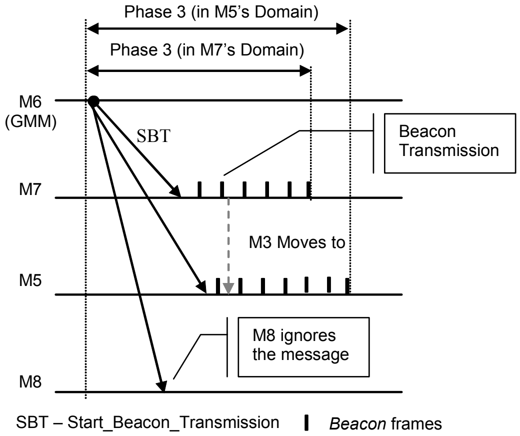


Figure 6.7 – Exchanged messages in Phase 3

Phase 4

After the end of the `Beacon` transmission, every wireless DMM (still holding the token) inquires all mobile wireless stations in order to detect if they are present in its domain, using `Discovery` messages. This period can also be referred to as the discovery sub-phase.

From this instant onwards, mobile wireless slaves are already capable of answering requests, but mobile wireless masters must still enter the new logical ring, using the standard PROFIBUS ring management mechanisms (described in Chapter 2). Since the routing table entries related to mobile wireless stations have been cleared, only when the BMs receive updated routing information (embedded on `Route_Update` messages), at the end of the IDMP, they may restart routing IDTs related to mobile wireless stations. Figure 6.8 depicts and example showing the sequence of events and the related messages exchanged during Phase 4, related to mobile wireless master M3.

The `Route_Update` messages are transmitted by the DMMs whenever they detect that a mobile station is ready to start operating; that is, after the entry of a master into the logical ring or after the detection of mobile wireless slave using `Discovery` messages.

When a mobile wireless station continues in the same domain, its presence is detected by the `Discovery` message and a `Route_Update` message is transmitted by the DMM before releasing the token (not shown in Figure 6.8). When a mobile wireless slave changes to another domain, the detection in the new domain is also made by the `Discovery` message. When a mobile wireless master changes to another domain, its detection is made by the update of the LAS and/or `GAP List` of the DMM of the new domain. After detecting the presence of mobile wireless stations, the DMM broadcasts a `Route_Update` message.

When a `Route_Update` message is received by a BM, it updates its routing tables according to the information contained in the message.

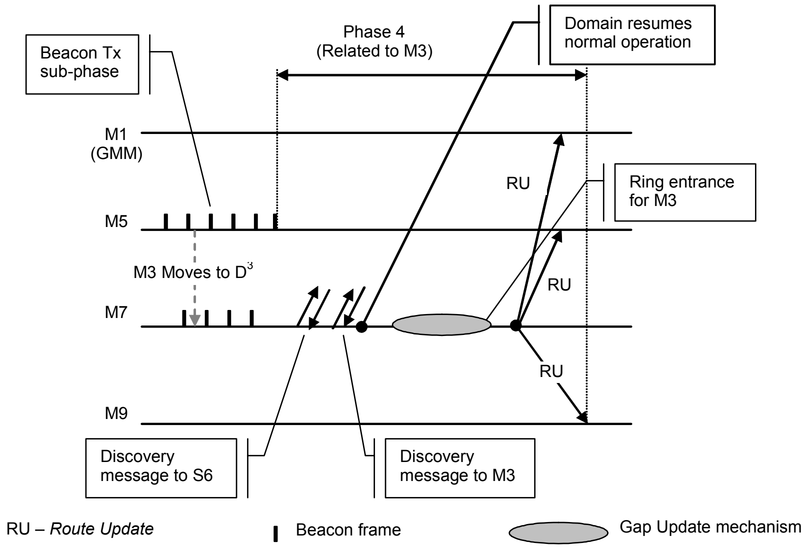


Figure 6.8 – Exchanged messages related to M3 in Phase 4

Finally, it is important to note that Phase 4 is asynchronous between domains, and, as in the case depicted in Figure 6.8, this phase starts at the end of the `Beacon` transmission in a wireless domain, and finalises when a mobile wireless station is operational in another or in the same domain.

6.2.4. Enabling the Mobility of Mobile Wired Domains

Mobile wired domains, as the one exemplified in Figure 4.6, may be the communication infrastructure used in devices like Automatic Guided Vehicles (AGVs) or Mobile Robots. The mobility of such devices is only possible if the associated wireless BM also supports the mobility related functionalities defined for mobile wireless master stations, i.e. this BM should be capable of assessing the quality of wireless radio channels and be able to switch between radio channels.

However, only the wireless BM is detected on the new domain (after joining the logical ring) and, consequently, the concerned domain DMM will send a

Route_Update message related to that single station. To complete the procedure, the necessary Route_Update messages containing the addresses of all stations belonging to the mobile wired domain must also be sent. In order to reduce the network traffic, each of these messages may embed data relative to several stations.

The stations belonging to mobile wired domains are signaled in the BMs as belonging to a wireless domain, avoiding in this way that wireless BM transmit Discovery messages related to such kind of stations.

6.2.5. Routing Tables Operation with IDMP

The routing tables defined in Section 5.2.2 must also contain the Station Type field, which is used in order to support the mobility of stations. Table 6.1 presents an example, which shows an excerpt of a RT regarding the network scenario presented in Figure 4.9 for BM M6 and M7.

Table 6.1 – Routing Table (example)

Destination	M6	M7	Station Type
M1	N	Y	F
M2	Y	N	F
M3	N	Y	MM
S6	Y	N	MS
…	…	…	…

The type field may assume the values {WrM , WrS, DRWlM DRWlS} for non-mobile stations, and the values {MWlM, MWlS, MWrM, MWrS} for stations with inter-domain mobility capability.

6.3. Example Scenario

In order to exemplify the operation of the IDMP, the network scenario depicted in Figure 6.3 is considered. For the sake of simplicity, it is considered that there is no additional traffic in the network, except for an IDT between master M2 and slave S7, an IADT between M2 and S5, the token passing, and the mobility-related messages. It is also considered that both M3 and S6 move between wireless domains during the evolution of the IDMP. Also for the sake of simplicity, bit rate and frame formats are identical in all domains. Figure 6.9, 6.10 and 6.11, show the timelines and the messages associated with the referred period of network operation.

The GMM starts the mobility procedure by broadcasting the Start_Mobility_Procedure message (SMP). When the BMs in bridges (B1, B2 and B3) receive this message, they stop accepting new IDTs from the stations belonging to their domains, but will complete all open IDTs.

BMs M4, M5, M6, M7 and M9 do not have any open IDT, thus they immediately transmit the messages Ready_to_Start_Mobility_Procedure {RSMP/M4, RSMP/M5, RSMP/M9}. Note that the RSMP message related to BM M7 is relayed internally between BM M7 and BM M6, and also that the message related to M6 is passed internally in these stations between the state machines controlling the DMM and the GMM.

BM M8 has an open IDT related to request M2.1, which is completed and deleted from the LOT upon the second repetition of request M2.1 (by M2). After, it sends the `Ready_to_Start_Mobility_Procedure` (RSMP/M8). Master M2 keeps trying to execute the same transaction, but BM M8 ignores, since it has already received the message `Start_Mobility_Procedure`, therefore it does not accept any new IDTs. Meanwhile, IADTs between M2 and S5 may still carry on.

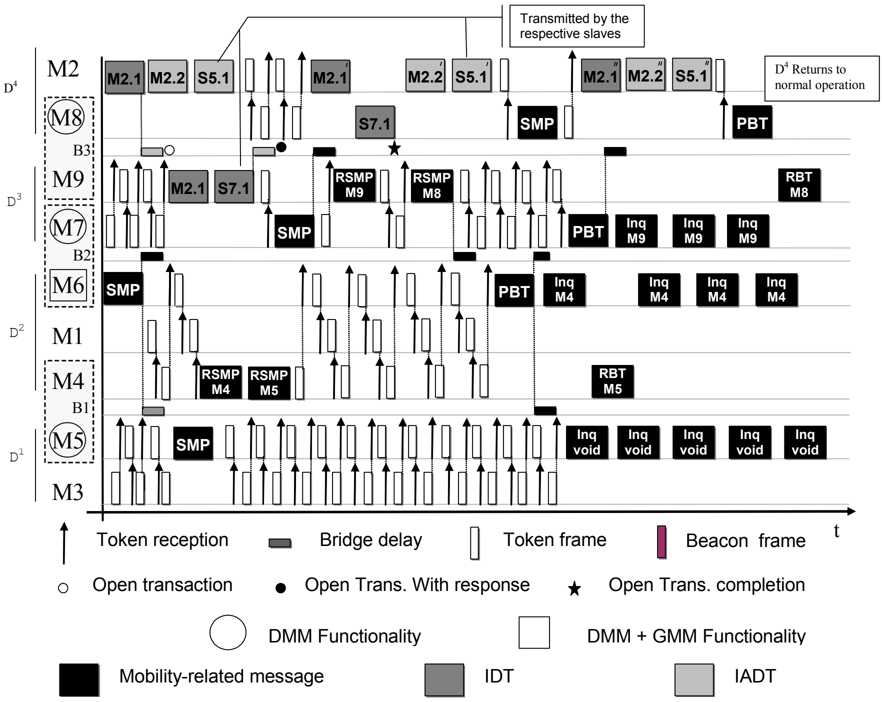


Figure 6.9 – Timeline for handoff procedure

After receiving the `Ready_to_Start_Mobility_Procedure` message from all BMs in the network, messages {RSMP/M4, RSMP/M5, RSMP/M9, RSMP/M8} and internally from M7 and from the DMM state machine of M6, the GMM broadcasts the message `Prepare_for_Beacon_Transmission` (PBT). When the DMMs M7, M8 and M5 receive that message, they wait until receiving the token. After that, the DMMs start the inquiry sub-phase. So, the messages signalling that the BMs have acquired the token (the `Ready_for_Beacon_Transmission` messages) are only transmitted to the GMM (M6), using the Inquiry service.

Message RBT/M5 is transmitted when M6 inquires M4. Similarly, message RBT/M8 is also transmitted by M9 when inquired by M7. Finally, the message related to

DMM M7 is transmitted internally, using the common functionality services provided by bridge B2.

Also note that BM M5 does not have any other bridges belonging to its domain, thus it sends Void frames in order to maintain the network activity. In terms of the mobility procedure, domain D^4 is a terminal domain, therefore after receiving the PBT message it can continue its normal operation.

After receiving the Ready_for_Beacon_Transmission message from all the DMMs in the network {RBT/M5, RBT/M7 and RBT/M8}, the GMM triggers the emission of Beacons by sending the Start_Beacon_Transmission message (SBT), as shown in Figure 6.10. This message triggers the emission of Beacon frames by the Structuring Intermediate System or by a bridge of the type Structuring and Linking Intermediate System (the case represented in the example).

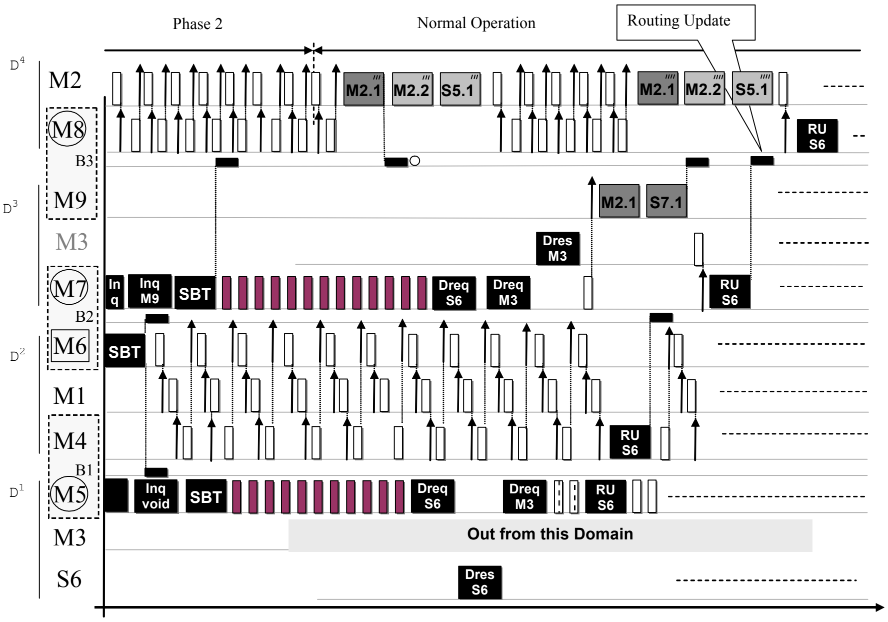


Figure 6.10 – Timeline for handoff procedure (cont.)

The starting time of this Beacon transmission is slightly different in the different domains, due to communication latencies between the GMM and the domain DMM. Also, the duration of the Beacon transmission must be different in the different domains, in order to insure that all mobile wireless stations have enough time to assess the quality of all radio channels. In this way, all domains will finish almost at about the same time. In Chapter 7, we will elaborate further on this.

In wired domains it is not necessary to transmit Beacon frames. Therefore, these domains can return to normal operation. Nevertheless, the bridges belonging to wired domains must relay the Start_Beacon_Transmission (SBT) message to other wireless domains.

Before the end of the Beacon transmission, mobile wireless master M3 and mobile wireless slave S6 switch to the radio channels of domain D^3 and domain D^1, respectively.

After the end of the Beacon transmission, the wireless DMMs M5 and M7 send Discovery messages {Dreq/S6, Dreq/S3} addressed to the mobile wireless stations in the network (M3 and S6) in order to detect if they are present in theirs.

From this point forward, slave S6 is capable of answering requests, but master M3 must first enter into the new logical ring, using the standard ring management procedures. This is illustrated in Figure 6.11.

Message RU/S6 is the Route_Update message related to station S6, but the message related to station M3 will only be sent when M3 effectively enters into the logical ring of domain D^3.

Master M2 has previously issued request M2.1''' (see Figure 6.10 for details), but it is only relayed by bridge B3 when BM M9 is again capable of relaying IDTs. That happens when BM M9 receives the token, after the end of the Beacon transmission. As it can be observed, intra-domain transactions in domain D^4 are possible during this period.

When M3 enters into the new wireless domain, it detects that it was taken out of the ring and goes into the Listen Token state. M3 will only be able to enter the new logical ring when its predecessor station (M9) starts the Gap Update mechanism and subsequently passes the token to M3.

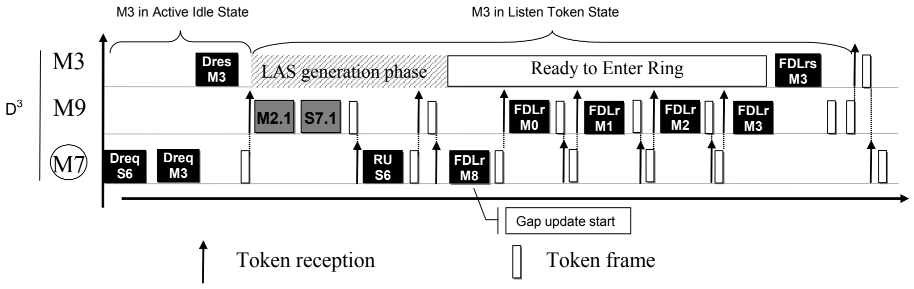


Figure 6.11 – M3 entrance into the logical ring

Figure 6.11 shows in detail how the entry of station M3 is performed. As it can be seen, after the channel switching, station M3 is still on the Active Idle state, so, it can return an answer (Dres/M3) to the Discovery message (Dreq/M3). After that, M3 detects that its predecessor station did not pass it the token, and therefore goes into the Listen Token state, where it re-generates its List of Active Stations (LAS) during two complete token rounds. In this state, M3 does not answer any requests addressed to it. After this, M3 is ready to enter into the logical ring and it is able to reply to any FDL_Request_Status frame (used by the Gap Update mechanism) indicating its readiness.

M9 uses the Gap Update mechanism in order to include M3 on its domain logical ring, thus it sends FDL_Request_Status requests {FDLr/M0, FDLr/M1 and FDLr/M2} addressed respectively to stations with addresses 0, 1 and 2 (considering that station M9 HSA is equal to 9). Finally, M9 sends a FDL_Request_Status request (Dreq/M3), which is addressed to M3, and replies with the standard PROFIBUS Ready to Enter Logical Ring message. Subsequently, M9 passes the token to M3, which only accepts it at the second retry. To make the entry procedure faster, master stations must have a low Gap Update factor.

6.4. Details on the IDMP Agents Implementation

6.4.1. State Machine for the Global Mobility Manager

The operation of the GMM is based on the state machine depicted in Figure 6.12. For its description we are considering that there is a mobility timer used to trigger the IDMP in a periodic fashion.

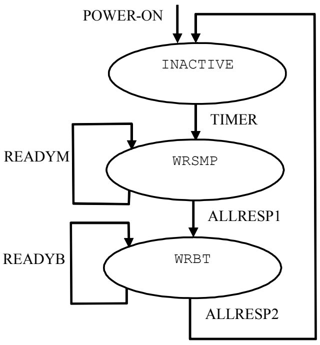


Figure 6.12 – State machine for the Global Mobility Manager

At power on, the GMM enters into the INACTIVE state, and the mobility timer is loaded with the Mobility Procedure period, which depends on the dynamics of the mobile wireless stations. When the mobility timer reaches zero (transition TIMER) the GMM state machine evolves to WRSMP state (Wait_Ready_to_Start_ Mobility_Procedure message) and the GMM sends the Start_Mobility_ Procedure message.

In the WRSMP state, the GMM receives Ready_to_Start_Mobility_ Procedure messages from all the network bridges (transition READYM). It will only evolve to WRBT (Wait_Ready_for_Beacon_Transmission message) when all bridges had replied (transition ALLRESP1) and then it sends the Prepare_for_ Beacon_Transmission message.

In the `WRBT` state, the GMM receives `Ready_for_Beacon_Transmission` messages from the network DMMs (transition `READYB`). When all DMMs have replied, the state machine returns into the `INACTIVE` state, and the GMM sends the `Start_Beacon_Transmission` message.

6.4.2. State Machine for the Domain Mobility Manager

The DMM is responsible for retaining the token, for of the *Inquiry* service and for the transmission of `Beacons` (only in a wireless domain). This functionality can be present in any type of resident or wired master station, but for a faster performance (in most cases) it should be located in a BM.

The DMM state machine (Figure 6.13) goes into the `INACTIVE` state after power-on. In this state, no mobility related functions are performed by the DMM.

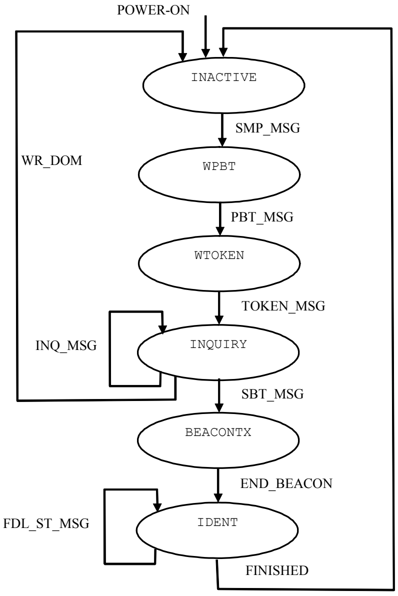


Figure 6.13 – State machine for a Domain Mobility Manager

Transition `SMP_MSG` is triggered when the DMM receives the message `Start_Mobility_Procedure`, and passes to state `WPBT` (`Wait_Prepare_for_Beacon_Transmission`), where the DMM waits for the reception of the `Prepare_for_Beacon_Transmission` message. This message triggers the transition to `WTOKEN` (Wait Token) state, transition `PBT_MSG`. In this state, the DMM waits until receiving the token from its predecessor. When it receives the token (transition `TOKEN_MSG`) the DMM retains the token and sends the `Ready_for_Beacon_Transmission` message to the GMM. Following this, the

DMM only uses the *Inquiry* service in order to exchange mobility-related messages with the bridges in its domain.

This service is needed in order to guarantee that DMMs in all domains are able to communicate with the GMM with minimal delays. Nevertheless, if a DMM does not have any other bridges belonging to its domain, then if the domain is not wireless it can resume normal operation. In the case of a wireless domain, then it transmits Void frames in order to maintain the network activity. This behaviour avoids any time-out errors, due to the absence of network activity, by the other stations belonging to that domain.

When the Start_Beacon_Transmission message arrives at the DMM (transition SBT_MSG) the DMM starts transmitting Beacon frames for a certain duration of time. When this period ends, the DMM will try to detect if any mobile wireless station is present in its domain by inquiring them using FDL_Request_Status frames (transition FDL_ST_MSG).

When a DMM is responsible for a wired domain it does not transmit any Beacon frames and thus it passes from the INQUIRY state directly to the INACTIVE state (transition WR_DOM).

6.4.3. DMM MAC State Machine

The PROFIBUS DLL state machine controls the operation of the PROFIBUS MAC protocol. It defines the initialisation of a station, its entrance into the logical ring, the token passing and the message cycle. Additionally, it also controls some functionalities related to the GAP Update procedure and to the management of errors.

The IDMP is designed in order to keep the number of modifications small, thus most of its functionalities can be implemented as an independent module above the DLL or at the Physical Layer level (like the channel assessment functionalities required by the mobile wireless stations). Nevertheless, some functionalities related to the DMMs must be implemented at the MAC sub-layer.

Figure 6.14, depicts the changes required to the PROFIBUS DLL of a BM in order to support the functionalities of a DMM. In this figure, only part of the original PROFIBUS State machine is illustrated.

In PROFIBUS a station enters into the Use_Token state after receiving the token. In this state the station is capable of performing at least one high-priority message cycle. After every transmitted request, the station enters into the Await_Data_Response state, where it waits for the reception of a response. When a response is received or at the expiration of the Slot Time and there are more messages to send on the station output queue, the station returns to the Use_Token state, otherwise the station evolves to the Pass_Token state.

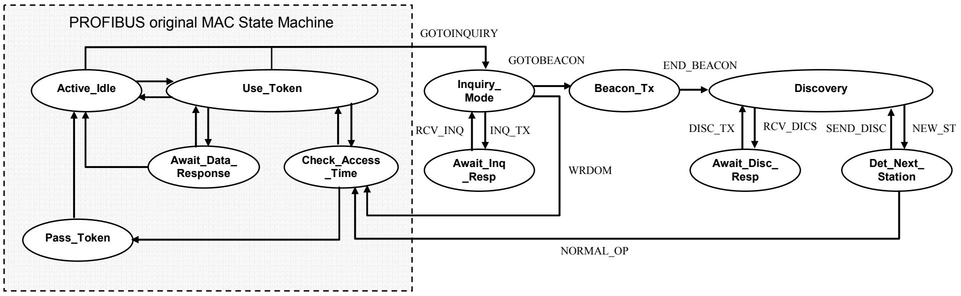


Figure 6.14 – DMM MAC State Machine

The station enters into the `Check_Access_Time` state, if at the beginning of the `Use_Token` state there are no pending high-priority message cycles to be performed; otherwise it always enters into this state before transmitting a request. In this state, the station computes its remaining token holding time and only performs a new transaction, if its value is greater than zero, otherwise it evolves to the `Pass_Token` state.

In the `Pass_Token` state the station passes the token to its successor and the state machine evolves to the `Active_Idle` state, during which the station waits for the token or for the reception of a request addressed to it.

The evolution of the state machine to the new states is partially controlled by the evolution of the state machine related to the DMM (Figure 6.13). So, a station enters into the `Inquiry_Mode` state at the reception of the token or when the station is in the `USE_TOKEN` state and after the DMM state machine had evolved into the `INQUIRY` state (transition `GOTOINQUIRY`).

In this state the station sends `Inquiry` frames, addressed to the domain bridges, and waits, by evolving to the `Wait_Inq_Response` state for any mobility related message sent by the addressed BM. The station evolves from the `Wait_Inq_Response` state (transition `RCV_INQ`) to the `Inquiry_Mode` state when it receives a response or when the `Slot Time` expires.

From `Inquiry_Mode` state a wireless DMM evolves to the `Beacon_Tx` state when the DMM state machine has also evolved to the `BEACONTX` state (transition `GOTOBEACON`). Contrarily, a wired DMM may evolve into the `Check_Access_Time` state and return to normal operation (transition `WRDMM`).

In the `BEACONTX` state the station transmits `Beacons` for a pre-configure amount of time, when it ends (`END_BEACON` transition) the station evolves into the `Discovery` state.

In the `Discovery` state the station sends `Discovery` message, addressed to the mobile wireless stations trying to detect the stations which belong to its domain. After entering this state the station evolves into the `Det_Next_Station` state (transition `SEND_DISC`) where the next `Discovery` frame is assembled, and the station evolves again into the `Discovery` state (transition `NEW_ST`) in order to transmit the message. When there are no more mobile wireless stations to inquiry, the DMM returns to normal operation by evolving into the `Check_Access_Time` state (`NORMAL_OP`).

BMs without DMM functionalities do not need any changes to its DLL, its main difference is related to the handling of the Inquiry messages, which must be replied with mobility related messages only.

6.4.4. Mobility-related Bridge Master Functionalities

The bridge's role during the IDMP is essentially in ensuring that there are no pending IDTs during the mobility procedure, and in the relaying of mobility-related messages (when the DMMs are on the INQUIRY state).

So, at power-on (Figure 6.15) the BM goes into the INACTIVE state, where it can operate normally relaying IDTs. In this state the BM can update its List of Active Stations, Live List or GAP List and consequently its routing table according to the changes on the configuration of the system (transitions LAS_C, LL_C and GAP_C). These transitions also trigger the broadcast of a Route_Update message.

Also, when the bridge receives a Route_Update message, it updates the routing tables and forwards the message.

When the bridge receives the Start_Mobility_Procedure message (transition SMP_MSG) it goes into the WIDT_END (Wait Inter-Domain Transactions End) state where the bridge waits until finalising all of its open IDTs contained in the LOT. Also, in this state the BMs will not accept new IDTs.

The completion of an IDT triggers the transition IDT_FINISHED. When all IDTs have been completed, the bridge enters into the WINQUIRY (Wait Inquiry message) state (transition ALL_IDT_FINISHED). On the WINQUIRY state the bridge only communicates with its domain DMM using the *Inquiry* service. In this state, when the bridge receives an Inquiry frame and it has the response (transition RESP) a response is sent, when it does not, no response is sent (transition NO_RESP).

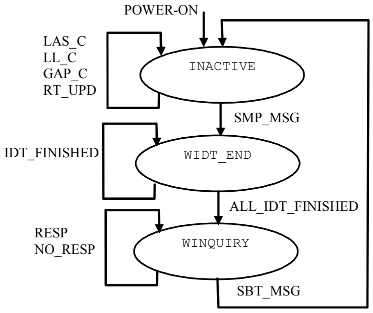


Figure 6.15 – State Machine for the mobility related functionalities in a bridge master

When the Beacon transmission starts, the bridge returns into the INACTIVE state and clears the entries related to mobile wireless stations in its routing table (transition SBT_MSG). Thus, all bridges must know the addresses of all mobile wireless stations in the system. From time onwards, the bridges are capable of relaying IDTs, if requested. Obviously, IDTs related to mobile wireless stations will only be relayed when the bridge receives the related `Route_Update` messages.

6.5. IDMP Messages

To reduce the costs and complexity of implementing the IDMP, this procedure is based on standard features offered by PROFIBUS. Therefore, all mobility-related messages use standard “Frames of Fixed Length with Data Field”, addressed to a specific SAP, e.g. 55, that handles the mobility procedure. Table 6.2, synthesises these messages.

Since most of the messages are sent in broadcast mode, thus not requiring any response, the frames are coded using high-priority SDN frames. Therefore, the `FC code` value of most protocol message, is set to 6. The field `Mobility Code` (MC) codes the type of operation that must be performed when the destination station receives the frame.

For the `Beacon` we propose to use the same type of message used in the RFieldbus system, which is described in (Rauchhaupt, 2003). This message must have a specific format, which allows the mobile wireless station to evaluate the radio channel quality.

Table 6.2 – Format of the Handoff Procedure protocol messages (requests)

	Frame Header				Frame Data			
Frame	**SD**	**DA**	**SA**	**FC**	**DAE**	**SAE**	**MC**	**Data**
Start_Mobility_ Procedure (SMP)	SD3	127	GMM	6	55	55	1	-
Ready_to_Start_Mobility _Procedure (RSMP)	SD3	GMM	Bri.	6	55	55	2	-
Prepare_for_Beacon_ Transmission (PBT)	SD3	127	GMM	6	55	55	3	-
Ready_for_Beacon_ Transmission (RBT)	SD3	GMM	DMM	6	55	55	4	-
Start_Beacon_ Transmission (SBT)	SD3	127	GMM	6	55	55	5	-
Route_Update (RU)	SD3	127	Bri.	6	55	55	6	Station Addrs
Inquiry	SD3	BM	DMM	13	55	55	7	-
Void	SD1	DMM	DMM	6	▨	▨	▨	▨

The `Inquiry` message is addressed to a BM on the DMM domain. This message is coded as SDR high service, since a response from the addressed BM is expected. In that case, the response to that service can only contain a mobility related message from the output queue of the addressed BM. Finally, when a wireless domain has only one bridge, and the bridge is also the DMM, it must transmit a `Void` message. We propose to use as a `Void` message, a “Fixed Length Frame without Data Field” addressed to the DMM.

The detection of stations in a domain after the transmission of `Beacons` is performed by the `Discovery` messages, which can be mapped onto standard PROFIBUS frames of the type `FDL_Request_Status`.

6.6. Summary

This chapter detailed the proposed Inter-Domain Mobility Procedure (IDMP) which is used to support the mobility of stations between different wireless domains in a hybrid wired/wireless PROFIBUS bridge-based network. The IDMP enables mobile wireless stations to move between different wireless domains, seamlessly, without errors or loss of frames. This chapter described the different phases required for the operation of the IDMP, the syntax of the exchanged messages, and some implementation details.

Chapter 7

Timing Analysis of the IDP and the IDMP

The support of real-time applications requires that the communication delays between devices are known and bounded. In this chapter, a timing analysis related to the Inter-Domain Protocol (IDP) and the Inter-Domain Mobility Procedure (IDMP) is proposed.

7.1. Introduction

A crucial factor on demonstrating the ability of the proposed hybrid wired/wireless PROFIBUS bridge-based network to support real-time applications is the provision of a timing analysis of its behaviour.

This chapter starts by presenting a Worst-Case Response Time (WCRT) analysis regarding IDTs which builds upon the results presented in Chapter 3 for the single logical ring approach and assumes that the IDMP is not active. Then, Section 7.3 uses the WCRT analysis regarding IDTs to elaborate a worst-case timing analysis related to the latencies associated with the different phases of the IDMP.

7.2. WCRT analysis for IDT transactions

PROFIBUS DLL defines 4 types of services for the exchange of data (for details the reader is referred to Chapter 2). These services can be classified in two types, which are relevant from the point of view of the timing analysis to be presented:

- when a transaction involves request and response frames – PROFIBUS `Send Data with Reply` (SDR) and `Send Data with Acknowledge` (SDA) services;
- and when a transaction only involves a request frame – PROFIBUS `Send Data with No acknowledge` (SDN) service;

In the following subsections both these two types of IDTs are analysed separately.

7.2.1. WCRT Analysis of IDTs Based on SDA or SDR Services

One of the characteristics of the Inter-Domain Protocol (IDP) is that the initiator periodically repeats a request until receiving a response. Consequently, the WCRT for a message stream in such conditions mainly depends on the message stream period. Figure 7.1 depicts a scenario where that fact becomes obvious. In that scenario, the response time for the represented message stream is equal to $2\times T_i^k+Rslr_i^k$, where T_i^k is periodicity of message stream i from master k and $Rslr_i^k$ is the response time on a Single Logical Ring (SLR) PROFIBUS network, which can be calculated by Eq. (3.1) or Eq. (3.4).

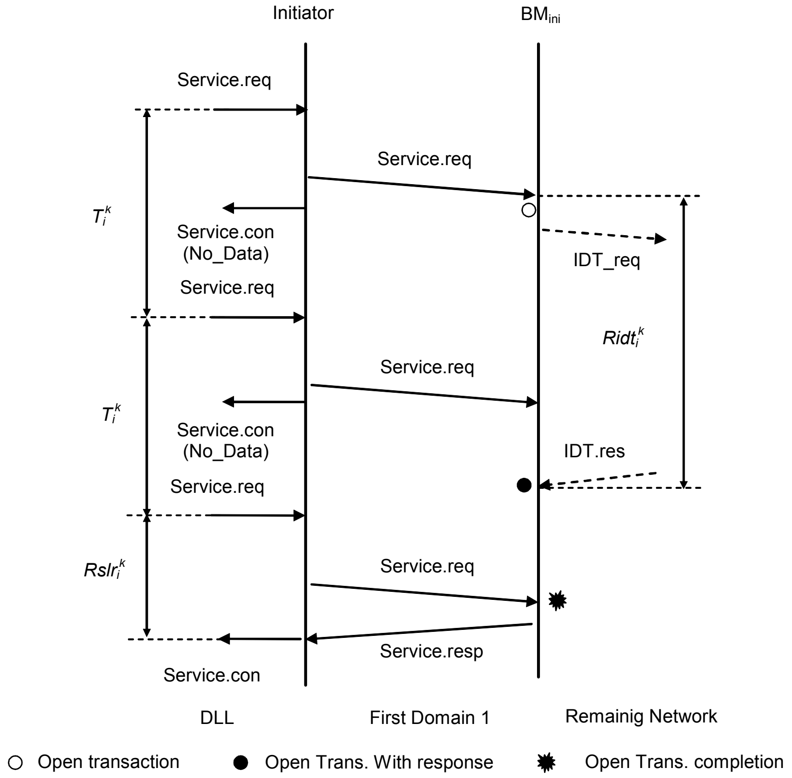


Figure 7.1 – IDT timing example

A_i^k is the maximum number of attempts performed by the initiator (master k) until receiving a valid response from BM BM_{ini}, related to message stream i. Consider also, that the last request from the initiator, the one that obtains the actual response, requires a WCRT of $Rslr_i^k$. Then, the WCRT for a message stream i from master k, on a MLR network ($Rmlr_i^k$), can be computed using the following formulation:

$$Rmlr_i^k = A_i^k \times T_i^k + Rslr_i^k \qquad (7.1)$$

The maximum number of attempts (A_i^k) depends on the delay experienced by the IDT, from the reception of the request at the BM_{ini} until the arrival of the respective response ($Rbmi_i^k$). To obtain A_i^k we must consider the worst-case situation on the side of BM_{ini}, that is, when the minimum amount of time elapsed between the reception of the first request and the request at which BM_{ini} is able to provide the actual response. Figure 7.2 depicts an example of such a combination of events.

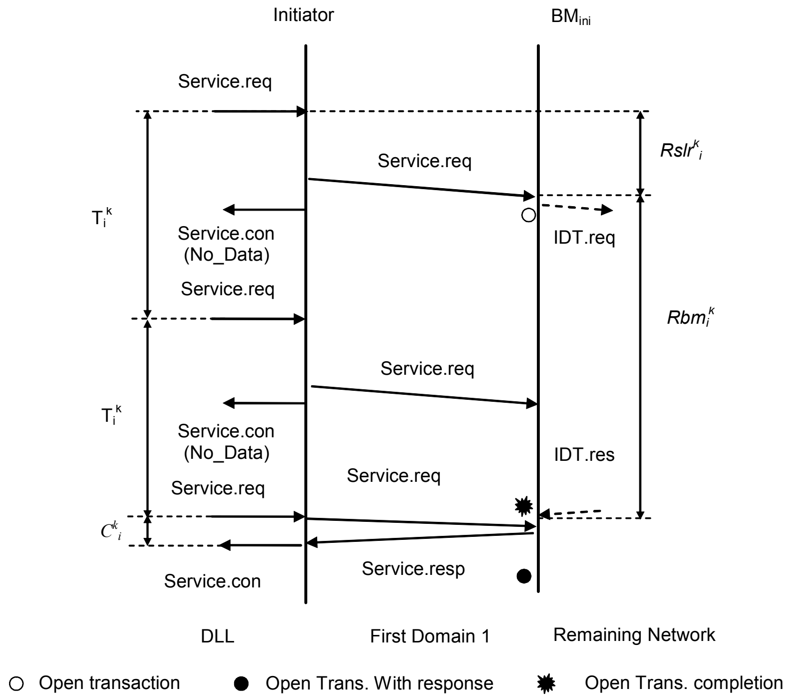


Figure 7.2 – IDT timings example (worst-case situation)

This situation occurs when the first request, the one that initiates the IDT, arrives at BM_{ini} delayed by its worst-case ($Rslr_i^k$), and the last request, the one that obtains the response arrives at BM_{ini} delayed by the best case (C_i^k). In this situation $Rslr_i^k + Rbmi_i^k < A_i^k \times T_i^k + C_i^k$. Therefore, since A_i^k must be an integer, its value can be obtained as follows:

$$A_i^k = \left\lceil \frac{Rslr_i^k + Rbmi_i^k - C_i^k}{T_i^k} \right\rceil \tag{7.2}$$

For the calculation of $Rbmi_i^k$, analysis can be adapted from the P-NET networks case (Tovar, 1999) and from (Ferreira, 2002) (which describes a similar MLR architecture).

Consider again the network scenario depicted in Figure 4.9, and a message stream S_1^{M1} between master M1 and slave S4. In this case, BM_{ini} will be the Bridge Master (BM) M5, and BM_{res} will be BM M8. To obtain the WCRT for an IDT transaction it is necessary to account for all the delays experienced by the IDT on the BMs, which depends on the number of streams processed by them, and on the traffic conditions on their respective domains.

Thus, using the WCRT analysis for a SLR PROFIBUS network, the following equation allows the calculation of $Rbmi_1^{M1}$ for the example outlined (between M1 and S4):

$$\begin{aligned}&Rbmi_1^{M1} = nh^{M1} \times T_{cycle}^{D^3} + (Creq_1^{M1})^{D^3} + \phi + \\ &nh^{M8} \times T_{cycle}^{D^4} + (Ch_1^{M1})^{D^4} + \phi + \\ &nh^{M9} \times T_{cycle}^{D^3} + (Cresp_1^{M1})^{D^3}\end{aligned}$$

nh^x is the number of high-priority message streams processed by a BM x. $T_{cycle}{}^d$ is the token cycle time in network domain d. $(C_{req}{}^{M1}{}_1)^d$ is the duration of the request frame in a domain d. The symbol ϕ is the Internal Forwarding Delay of the bridge, which represents the time needed by a BM, after receiving a request, to process the frame and forward it to the other BM. $nh^{M7} \times T^{D3}{}_{cycle}$, $nh^{M8} \times T^{D4}{}_{cycle}$ and $nh^{M9} \times T^{D3}{}_{cycle}$ are, respectively, the queuing delays on BMs M7, M8 for the inter-domain request frame, and on M9 for the inter-domain response frame. $(Ch_1{}^1)^{D4}$ is the transaction duration time between M8 and S4. $(C_{req}{}^{M1}{}_1)^{D3}$ and $(C_{resp}{}^{M1}{}_1)^{D3}$ are, the latency of the request frame in domain D^3, and the latency of the response frame in domain D^3, respectively.

In this analysis it is assumed that, at the reception of any IDF related to message stream $S_1{}^{M1}$, a BM has queued the maximum number of high-priority message streams relayed by it.

To obtain a general formulation, the analytical model defined in Chapter 4 can be used. In this formulation, b is the number of bridges between the initiator and the responder. $\Omega_{res}[1]$ represents the BM_{ini}, located in the initiator domain. $\Omega_{req}[b]$ represents the BM_{res} which is attached to the same domain where the responder is located, thus it will execute a complete transaction (including a request and a response). The network domains are numbered from 1 to b + 1, being the first domain of the IDT communication path (the one of the initiator), domain number 1 and the last domain (the one of the responder) numbered as $b+1$. Then, $Rbmi_i^k$ can be obtained as follows:

$$\begin{aligned}Rbmi_i^k &= \sum_{f=1}^{b-1}(nh^{\Omega_{req}[f]} \times T_{cycle}^{\Omega_{req}[f]} + (Creq_i^k)^{f+1}) + b \times \phi \\ &+ b \times \phi + \sum_{f=2}^{b}(nh^{\Omega_{res}[f]} \times T_{cycle}^{\Omega_{res}[f]} + (Cresp_i^k)^f) + \\ &+ nh^{\Omega_{req}[b]} \times T_{cycle}^{\Omega_{req}[b]} + \left(Ch_i^k\right)^{b+1}\end{aligned} \qquad (7.3)$$

In this equation, $(Creq_i{}^k)^d$ and $(Cresp_i{}^k)^d$ are the latencies associated to the transmission of a the request or a response on a network domain d, respectively.

It is also possible to rewrite Eq. (7.3) in a more compact format, as follows:

$$Rbmi_i^k = \sum_{f=1}^{b} Rslr_i^{\Omega_{req}[f]} + \sum_{f=2}^{b} Rslr_i^{\Omega_{res}[f]} + 2 \times b \times \phi \qquad (7.4)$$

$Rslr_i{}^{bm}$ is the worst-case response time for an IDT for a message stream i from master k, when the IDF is transmitted by a BM bm on the IDreq or IDres communication path. This value can be calculated by any of the SLR WCRT formulations described in Chapter 3, which are represented by Eq. (3.1) or Eq. (3.4).

It is important to note that the IDP defines different frame formats for the frames exchanged between bridges, and this aspect must be taken into account when calculating transaction latencies on each domain, as also the different frame format formats used by the wired and wireless physical layers.

At this point it is important to note that for the correct behaviour of the proposed IDP error control procedure proposed in Chapter 5, the value of the IDT *Error Handling Timer* must be set to a value larger than maximum $Rbmi_i^k$ of all message streams in the system.

7.2.2. WCRT Analysis of IDTs Based on the SDN Service

In PROFIBUS the SDN service is used to transmit a frame between an initiator to another station (unicast), to a group of stations (multicast), or to all stations in the network (broadcast). Since this kind of service is unconfirmed, then the IDP specifies that the system BMs just forward the frame to other domains without modifications.

As in the case of the SDR and SDA services, it is possible to provide a worst-case time bound for IDTs based on the SDN service involving just two stations. The cases of multicast and broadcast are not treated in this subsection.

The worst-case time required by a request from a message stream i, to go from a master k to another station w ($Ru_i^{k\to w}$), can be obtained by adapting Eq. (7.3) as follows:

$$Ru_i^{k\to w} = Rslr_i^k + \sum_{f=1}^{b} Rslr_i^{\Omega_{req}[f]} + b\times\phi \qquad (7.5)$$

In this equation, the u in $Ru_i^{k\to w}$ stands for unicast transaction. The first term ($Rslr_i^k$) in this equation represents the latency on master k's domain, and the other terms represent the latency associated with the relaying by the bridges.

Note, however, that if station k is a BM, then Eq. (7.5) needs re-formulation. As an example, consider the network depicted in Figure 7.3, and the following two transactions: a transaction S_1^{M2} that involves BM M2 and slave S21, and a transaction S_1^{M3} that involves BM M3 and slave S23. For transaction S_1^{M2}, the first leg of the path is between M2 and M3, with a delay equal to ϕ, thus the first station in the path, which transmits the message through the network, is M3. For transaction S_1^{M3}, the message is transmitted directly into domain D^2. This situation is particularly important for the timing analysis of the mobility procedure, since most of the messages related to the IDMP are transmitted in unicast or broadcast modes.

It is also possible that the destination station is a BM, e.g. M5. In that situation the message would be received by M4 and passed to M5.

Thus, in both cases, Eq. (7.5) requires some adaptations. If the transaction is similar to transaction S_1^{M3}, then the first station to transmit the request is the transaction initiator itself. If the transaction is similar to transaction S_1^{M2}, then the first station to transmit the request is the BM on other side of the bridge, and the message is delayed by ϕ before being queued. If the destination is a BM not directly connected to the last domain where the message is transmitted then, the message is also delayed by ϕ before being received by the destination BM. Thus, Eq. (7.5) can be rewritten as follows for the case when the transaction initiator is a BM:

$$Ru_i^{k\to w} = Rslr_i^{k'} + \sum_{f=1}^{b} Rslr_i^{\Omega_{req}[f]} + (b+di+df)\times\phi \tag{7.6}$$

k' is the first BM to transmit the request. This can be master *k* itself, when it is directly connected to the first domain in the path, or can be the BM on other side of the bridge if master *k* is not directly connected to the first domain in the path. *di* is equal to 0 if the initiator is a master station or if the initiator is a BM directly connected to the first domain in the path for message stream *i*. *di* is equal to 1 if the initiator is a BM not directly connected to the first domain in the path. *df* is equal to 0 if the destination station is a master, a slave or a BM directly connected to the last domain in the IDT Communication Path. *df* is equal to 1 if the destination station is a BM not directly connected to the last domain where the message is transmitted. Note that $\Omega_{req}[1]$ is the first BM (excluding the BM belonging to the initiator's bridge) that transmits the request message in a domain.

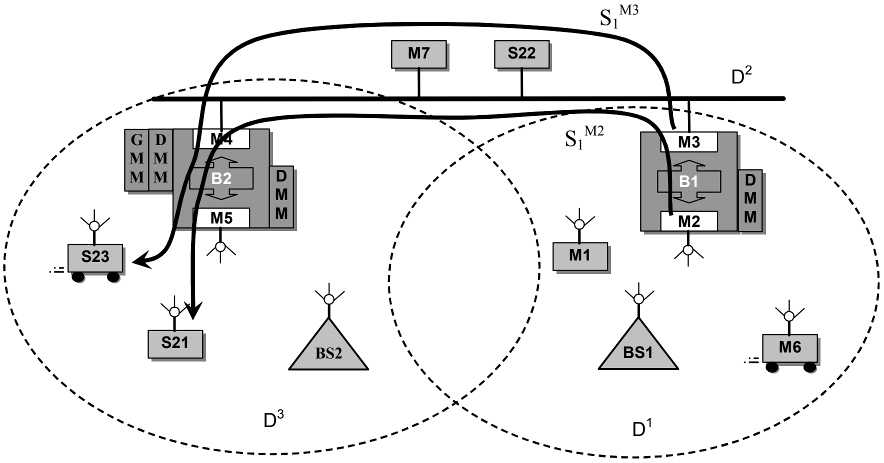


Figure 7.3 – Unicast IDT with a BM as initiator

7.2.3. Working on Reducing the Pessimism

The formulation presented in the previous section has a certain level of pessimism, which is inherent to considering the simultaneous occurrence of a number of worst-case situations. Although this may be a fate inherent to all guaranteed approaches based on worst-case scenarios, particularly in the case of distributed event-driven systems, it is important to investigate whether there is room for some improvements.

In this section we elaborate a bit further on this direction.

Eventually, one of the main sources of pessimism resides in the assumption made on considering that, in the worst-case, only one high-priority message can be processed by a master at each token visit (on the analysis proposed in (Tovar and Vasques, 1999a)). On the analysis proposed by (Cavalieri *et al.*, 2002) the main source of pessimism is related to assuming that all message transactions have the same size (equal to the longest message transaction).

Also, it may not be a negligible source of pessimism the assumption that all message streams relayed by a BM will be ready for transmission at the same time.

While fighting against the first would eventually collide with a basilar (from the real-time perspective) approach for handling such type of real-time guarantees in PROFIBUS networks, the latter probably deserves a second thought. In fact, being able to better model the maximum number of message streams that can be simultaneously queued by the BMs, may strongly impact the values for $Rbmi_i^k$ (Eq. (**7**.3)).

For the calculation of $Rbmi_i^k$, we assumed a worst-case situation at each bridge, in which all message streams relayed by a BM could be queued for transmission just prior to the instant when a frame related to a message stream *i* from master *k* arrives at the BM. In fact, that assumption can be somehow relaxed. On a dual port bridge, the messages arriving at the bridge are received in sequence by one of the bridge ports, hereafter called the input BM. At the same time, these messages can be transmitted by the other bridge port – the output BM of the bridge. Consequently, in some cases, when a frame from message stream S_i^k is queued on the output BM, the output queue will not, simultaneously, have frames from all message streams.

Figure 7.4 supports further intuition on this. The illustrated example assumes the network structure presented in Figure 4.9, and describes the sequence of events for transactions between master M3 and slave S2, related to 4 message streams, which are represented in Figure 7.4 by S3.1, S3.2, S3.3 and S3.4, respectively. We are also assuming that no other traffic exists in the network. For convenience, messages transmitted by S2 are depicted in the same line of BM M4.

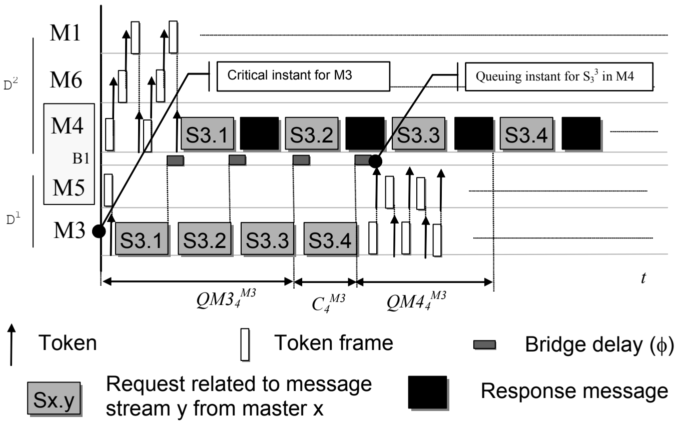


Figure 7.4 – IDTs frames arriving at a BM

In this example, it is considered that all request messages are queued just prior to the reception of the token by M3 (the critical instant for M3). As it can be observed, when a frame from message stream S_4^{M3} is ready for transmission by BM M4, the response frame related to message streams S_1^{M3} and S_2^{M3} had already been obtained by bridge B1. In the figure, the queuing delay for S_4^{M3} in master M4 ($QM4_4^{M3}$) is shown, as also the queuing delay in BM M4 ($QM4_4^{M3}$). $QM4_4^{M3}$ depends on the number of frames queued on M4 (in this case, 2), at the arrival of a frame from message stream S_4^{M3} and the ongoing transactions on wired domain 1.

What is added here for computing the worst-case response time, is that it should not be needed to consider that, in all cases, all message streams relayed by BMs are on their output queues simultaneously.

Therefore, we could introduce $nh'_{IDT}{}^{bm}$ to denote the maximum number of IDT transactions simultaneously queued by a BM *bm* (note the impact of *nh* in Eq. (**7**.3)).

The analysis proposed in (Tovar and Vasques, 1999a) guarantees that at least one high-priority message is dispatched per token visit, thus it is guaranteed that the output queue of a BM is reduced by one element at least once every token visit.

To devise a general formulation to this problem. Consider a bridge constituted by a BM *k* and a BM *l*. BM *k* receives the incoming traffic from its domain – the input domain, and BM *l* forwards the traffic to another domain – the output domain.

The incoming traffic can be characterised as follows:

- all message streams related to IDT that use BM *k* arrive in sequence;
- we assume that all messages have the same size (*Cin*), equal to the minimum size of the input message streams;
- all messages arrive at BM *k* with minimal separation.

The traffic forwarded by BM *l* can be characterised as follows:

- just prior to receiving the first message concerning input traffic, there is a transmission opportunity;
- the remaining transmitting opportunities are separated by the cycle time of the output domain (t^{out}_{cycle}).

Figure 7.5, depicts, on a simplified timeline, the arrivals at BM M5 and the transmissions by BM M4, which illustrates the assumptions.

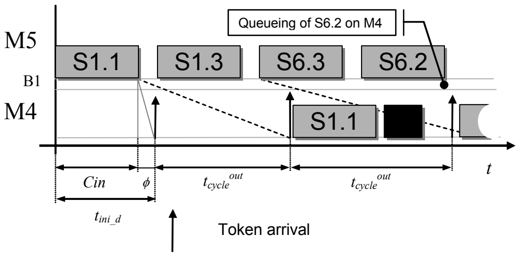


Figure 7.5 – Worst-case relaying scenario

The initial delay (t_{ini_d}) is equal to $Cin + \phi$. To obtain the number of messages which can be transmitted by the output BM, we propose the algorithm shown in Figure 7.6.

```
Cal_nh'_idt(Cin, nh_idt, tcycle_o, ϕ)
{
  // F: vector that contains if an IDT had been
  // forwarded by the output BM
  // Cin: equal to the minimum size of the input message streams

  ttotal = Cin × (nh_idt − 1); //Considered interval
  $t_{ini\_d}$ = Cin + ϕ;
  t = $t_{ini\_d}$ + tcycle_o; // inicial time
  nm = 0

  while t + Cin < ttotal {
     // messages available
     msg_av = floor((t -ϕ)/ Cin);
     if t > nm × Cin and F(nm) == 0 and nm ≤ msg_av then {
         F(nm) = 1;
         nm = nm + 1;
     }
     t = t + tcycle_o;
  }
  nh'_idt = nh_idt - nm
  return(nh'_idt)
}
```

Figure 7.6 – Algorithm for obtaining *nh'*$_{IDT}$

7.3. IDMP Timings

Throughout the progress of the IDMP, there are periods of time during which some network domains are inaccessible. That is the case of the periods corresponding to the Beacon transmission and inquiry sub-phase, where normal transactions, between any two nodes in the same domain, are not possible. Additionally, IDTs are disabled from the middle of Phase 1 until the end of the Beacon transmission. Thus, the mobility procedure affects the worst-case time for transactions depending on the location and type of the stations involved.

IADTs are disabled during the inquiry sub-phase, the Beacon transmission sub-phase and the station discovery sub-phase. Thus, in order to include the effect of the IDMP, the worst-case response time must be updated to $Rslr\_m_i^k = Rslr_i^k + t_{IADT\_dis}$. The *m* in *Rslr_m* denotes the inclusion of the mobility related delays. t_{IADT_dis} represents the time during which IADT are disabled in a domain. On the other hand, the impact of the mobility procedure on the WCRT for IDTs, is reflected on the way in which parameter A_i^k is obtained. In the remaining of this section a thorough characterisation of the IDMP timings is presented, which will then be used in Chapter 8 to devise a WCRT analysis for IADT and IDT in the proposed architecture, considering the influence of the IDMP.

7.3.1. Phase 1

The IDMP starts with the transmission, by the GMM, of the `Start_Mobility_Procedure` (SMP) message, which must be received by all BMs in the system. The worst-case time for the `Start_Mobility_Procedure` message to reach a BM *bm* is denoted as t_{SMP}^{bm}, and can be calculated considering an unicast IDT (Eq. (7.6)): $t_{SMP}^{bm} = Ru_{SMP}^{GMM \rightarrow bm}$. Note that since the `Start_Mobility_Procedure` message must be relayed by the bridges, for computing this time span it is necessary to include the SMP message as a message stream contending for transmission with the other message streams relayed by the BMs.

After receiving the `Start_Mobility_Procedure` message, the BMs stop accepting new IDTs from masters belonging to their domains. Nonetheless, they keep handling pending IDTs and, importantly, they keep handling IDTs originated in the other domains. After completing all pending IDTs, the bridges signal their new state by transmitting a `Ready_to_Start_Mobility_Procedure` message, addressed to the GMM. Figure 7.7 illustrates Phases 1 and 2 timings assuming the network scenario depicted in Figure 4.9. Note that although only the system DMMs are represented, the `Ready_to_Start_Mobility_Procedure` message must the transmitted by all BMs in the system.

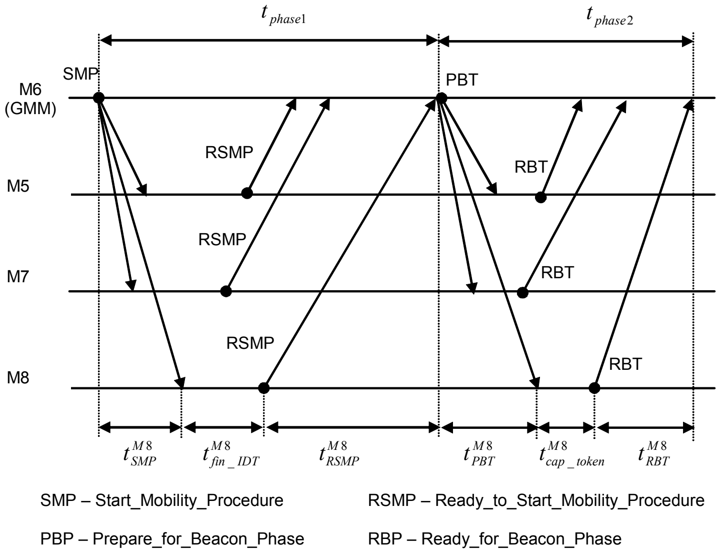


Figure 7.7 – Phase 1 and Phase 2 main events timings

The worst-case time until all pending IDTs are completed is different on the considered BMs and depends on the characteristics of the message streams served by the BMs. In this context we are considering that a BM serves a message stream when the message stream originates from a station on its domain and the BM is the first in the path (from the initiator to the responder).

To obtain that value, we assume the following conditions:

- all initial requests, related to the IDTs served by BM *bm*, arrive just before to the reception of the `Start_Mobility_Procedure` message, thus BM *bm*, has its maximum number of IDTs (nh^{bm}) simultaneously queued on its output queue;
- the corresponding IDT response arrives to BM_{ini} just after the transmission of the same request frame by the initiator.

In these conditions, the following equation gives the worst-case time until all IDTs are completed for a particular BM *bm*.

$$t^{bm}_{fin\_IDT} = \max_{S^k_i \in \Psi_{IDT}} \left\{ Rbmi^k_i + T^k_i + Rslr^k_i \right\} \tag{7.7}$$

Ψ_{IDT} refers to the set of message streams which are also IDTs served by BM *bm*. *k* represents a master which belongs to the domain where the BM *bm* is connected, and uses BM *bm* as the first BM in the path – BM_{ini}. $Rbmi^k_i$ is the response time, counting from the reception of the initial IDT request until the reception of the corresponding IDT response, by BM_{ini}, which can be calculated using Eq. (7.3).

After finalizing all IDTs, the BMs transmit the `Ready_to_Start_Mobility_Procedure` (RSMP) message (see Figure 7.7 for details). The worst-case time needed by this message to go from a BM *bm* to the system GMM (${t_{RSMP}}^{bm}$) can also be calculated using Eq. (7.6): ${Ru_{RSMP}}^{bm \rightarrow GMM}$.

Phase 1 only stops when all `Ready_to_Start_Mobility_Procedure` messages (coming from all DMMs) have been received by the GMM. Since the duration of Phase 1 is different for the diverse BMs, then its worst-case duration is equal to the maximum of: the time required by the `Start_Mobility_Procedure` message to reach a BM *bm*, added to the time required by BM *bm* to finalize its pending IDTs, added to the time required by the `Ready_to_Start_Mobility_Procedure` Message to reach the system GMM. The following equation holds:

$$t_{phase1} = \max_{\forall bm} \left\{ t^{bm}_{SMP} + t^{\mathrm{bm}}_{fin\_IDT} + t^{bm}_{RSMP} \right\} \tag{7.8}$$

Only at this point in time the GMM can proceed to Phase 2.

7.3.2. Phase 2

Phase 2 starts immediately after the end of Phase 1, when the GMM sends the `Prepare_for_Beacon_Transmission` (PBT) message. After receiving this message, and as soon as a DMM receives the token, it will retain the token and will not pass it to other masters in its domain. Following that, the DMMs send a `Ready_for_Beacon_Transmission` (RBT) message to the GMM and enter into inquiry sub-phase. In this sub-phase, the domain DMMs inquire, in sequence, their domain BMs, whether they have any `Ready_for_Beacon_Transmission` message available. The inquiry sub-phase helps in reducing the communication latency between the GMM and the DMMs, and keeps small the inaccessibility period of the network. When a BM, with or without domain management capabilities, receives the `Prepare_for_Beacon_Transmission` message, it will only be able to communicate using the Inquiry service, and it clears all its routing table entries related to mobile wireless stations. See Figure 7.7 for further intuition on the message and event sequence.

The worst-case time required for the `Prepare_for_Beacon_Transmission` message (time span denoted as ${t_{PBT}}^{dmm}$) to reach DMM *dmm* is given by ${Ru_{PBT}}^{GMM \to dmm}$. Note that during this time span there are no other IDTs going on. So, the only inter-domain traffic in the network is related to the `Prepare_for_Beacon_Transmission` message and, as a consequence, the BMs only queue messages related to the "branches" below it (remember that the network topology is tree-like). Using again Figure 4.9 as an example, and to illustrate this case, M4 would have to forward one message related to DMM M5, and M6 would have forward two messages, one related to DMM M7 and another related to DMM M8.

After receiving the `Prepare_for_Beacon_Transmission` message, the DMMs will have to capture the token on their respective logical rings. The worst-case time required until capturing the token (denoted as ${t_{\text{cap\_token}}}^{\text{dmm}}$) is equal to the worst-case token rotation time of the domain where the DMM *dmm* is located, ${T_{cycle}}^{dmm}$, which can be computed as explained in Chapter 3.

With the network operating in inquiry mode, the worst-case time required for the `Ready_for_Beacon_Transmission` message to go from the DMM *dmm* until the GMM can be computed as follows:

$$t_{RBT}^{dmm} = \sum_{x=0}^{b-1} (Rinq_{RBT}^{2x \to (2x+1)} + \phi) \tag{7.9}$$

where $Rinq_{RBT}^{x \to (x+1)}$ is the worst-case delay experienced by the `Ready_for_Beacon_Transmission` message when being transmitted from a BM x to another BM $x+1$, in the path to the GMM. For this formulation we assume that the BMs in the path, between DMM *dmm* and the system GMM are numbered as follows: {0, 1, 2, …, 2×*b*-1), where 0 refers to DMM *dmm* and 2×*b*-1 to the GMM. *b* is the number of bridges in the path. Since the GMM is also a DMM for its domain, then it is not necessary to transmit the `Ready_for_Beacon_Transmission` message in this domain.

To provide further intuition related to the inquiry mode, the reader is referred back to the example addressed in Figure 6.9.

The inquiry mode starts, in domain D^2, after the transmission of the `Prepare_for_Beacon_Transmission` message by M6 (the system GMM). In this mode M6 sends, repeatedly, the `Inquiry` message (denoted as Inq/M4) to BM M4. If it has any `Ready_for_Beacon_Transmission` message on its output queue then that message is transmitted. BM M4 must transmit a `Ready_for_Beacon_Transmission` message related to the DMM of domain D^1 (RBT/M5).

To calculate the worst-case latency of a transaction when the system is in the inquiry mode, we assume the following conditions about the network operation:

- the BMs only transmit mobility related messages, IDTs are disabled;
- the message to be transmitted arrives at a BM *a* just after the domain DMM has inquired BM *a*;

- at any given instant the maximum number of queued messages in a BM *bm* is equal to the number of bridges belonging to the branches under that BM.

In these conditions, the worst-case time needed to forward a message stored on a BM *a*, to another BM *b*, in the same domain, is given by the following equation:

$$Rinq_{msg}^{a\to b} = ((C_{Inq})^{dmm} + (C_{Ires})^{dmm}) \times (n_{iBMs})^{dmm} \times n_{msg}^{a} \tag{7.10}$$

where $(C_{Inq})^{dmm}$ is the worst-case latency associated with the `Inquiry` request message and $(C_{Ires})^{dmm}$ is the worst-case latency of the `Inquiry` response message on a domain (represented by its DMM). According to the IDMP protocol, a BM can only reply with mobility related message. Therefore, the maximum latency of this message is equal to the maximum latency of the messages presented in Table 6.2. In Eq. (7.10) $(n_{iBMs})^{dmm}$ is the number of BMs which are inquired by the DMM, and $n_{msg}{}^{a}$ is the maximum number of messages that may be stored in BM *a*. Just as an example, in the case a `Prepare_for_Beacon_Transmission` message is being transmitted from M4 to M7 then $Rinq_{PBT}^{M4\to M7} = ((C_{Inq})^{M7} + (C_{PBT})^{M7}) \times 2 \times 2 \cdot$

It is also necessary to analyse the case when the station transmitting the mobility related messages is the DMM itself. In this case it is assumed that:

- the DMM suspends the execution of the polling cycles with the remaining BMs in the domain, and transmits its mobility related messages;
- the message to be transmitted arrives at the DMM just after it had started a polling cycle with another station.

Therefore, given these assumptions, the following equation updates equation (7.10) when the message sender is the DMM:

$$Rinq_{msg}^{a\to b} = \begin{cases} ((C_{Inq})^{dmm} + (C_{Ires})^{dmm}) \times (n_{iBMs})^{dmm} \times n_{msg}^{a} & \text{, a is not a DMM} \\ ((C_{Inq})^{dmm} + (C_{Ires})^{dmm}) + (C_{msg})^{dmm} \times n_{msg}^{a} & \text{, a is a DMM} \end{cases} \tag{7.11}$$

To obtain the worst-case time span for Phase 2, the following analytical formulation may then be applied:

$$t_{phase2} = \max\left\{t_{PBT}^{dmm} + t_{cap\_token}^{dmm} + t_{RBT}^{dmm}\right\}, \forall\ \text{dmm} \tag{7.12}$$

7.3.3. Phase 3

After collecting all `Ready_for_Beacon_Transmission` messages from all the DMMs, the GMM starts the `Beacon` transmission sub-phase by broadcasting the `Start_Beacon_Transmission` (SBT) message. Upon receiving this message, the DMMs start emitting `Beacons`. In wired domains no `Beacons` are transmitted, and therefore stations in these domains may resume IADTs. Stations in a wired domain can execute IDTs with other wired stations, if the domains to which they are connected are able to complete those transactions, i.e. if they are wired domains.

Therefore, the duration of Phase 3 will be equal to the time needed by a DMM to receive the `Start_Beacon_Transmission` (SBT) message, added to the duration of the Beacon transmission sub-phase. Figure 7.8, depicts a timeline for the sequence of events during Phases 3 and 4.

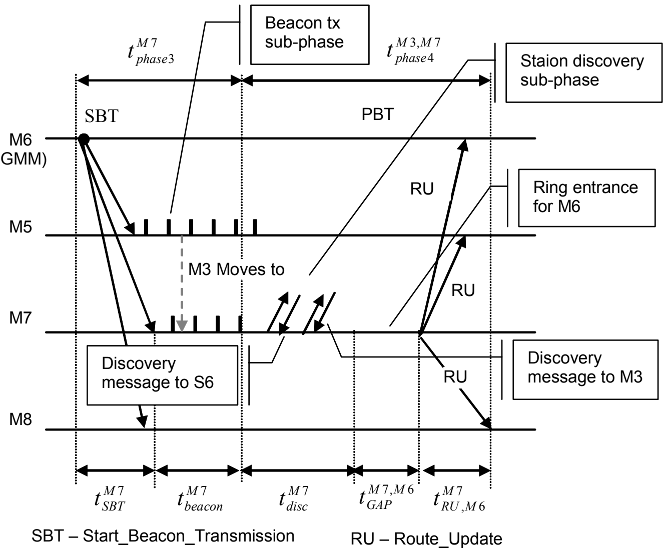


Figure 7.8 – Timeline for Phases 3 and 4

The worst-case time required by the Start_Beacon_Transmission message to reach a DMM *dmm* in the system is given by the following equation:

$$t^{dmm}_{SBT} = \sum_{x=0}^{b} Rinq^{2x \to 2x+1}_{SBT} + \phi \tag{7.13}$$

where *x* represents the list of BMs in the IDT Communication Path, from the GMM to a DMM *dmm*, which relay the Start_Beacon_Transmission message, similar to the formulation in Eq. (7.3). The only difference is that, in this case, station 0 represents the first BM in the IDT Communication Path that transmits the SBT message (which can be the GMM, if it is directly connected to a domain on the path do DMM *dmm* or not) and station 2×*b*-1 represents the DMM *dmm*. *b* is the number of bridges between the GMM and DMM *dmm*.

The duration of the Beacon transmission sub-phase ($t_{beacon}{}^{dmm}$) is a parameter that is setup individually on every domain. It must be set in a way that guarantees that every mobile wireless station has enough time to evaluate all the available radio channels. The reader is referred to Section 7.3.6 for details on how to calculate this parameter.

The worst-case duration of Phase 3, calculated for every wireless domain *d* (represented in the equation by its DMM), is given by:

$$t^{dmm}_{phase3} = t^{dmm}_{SBT} + C_{beacon} \times n^{dmm}_{beacon} \tag{7.14}$$

where, C_{beacon} is the worst-case latency associated with the transmission of a Beacon frame.

7.3.4. Phase 4

After the end of the Beacon transmission sub-phase, every wireless DMM (still holding the token) inquires all mobile wireless stations, using the Discovery message, in order to detect if they still belong to its domain or to detect new "entries" on its domain. After this, mobile wireless slaves are capable of answering requests, but new mobile wireless masters must still enter the logical ring using the standard PROFIBUS Gap Update mechanisms. After, the discovery sub-phase the DMMs send Route_Update messages containing the addresses of the mobile wireless slaves that moved and the mobile wireless slaves and master that are still in the DMM domains. Route_Update messages related to mobile wireless masters which moved are not sent after their entering in the logical ring.

To obtain the duration of the station discovery sub-phase, the following conditions must be assumed:

- a wireless domain DMM (still holding the token) will inquiry all mobile wireless stations, starting from the station with lower address;
- all mobile wireless station are on the same domain (the worst-case situation).

It follows that the worst-case duration of the station discovery sub-phase can be computed by:

$$t_{disc}^{dmm} = n_{mob\_stations} \times C_{dic}^{dmm} \tag{7.15}$$

where $n_{mob\_stations}$ is the number of mobile wireless stations (including masters and slaves), and C_{disc}^{dmm} is the worst-case latency associated with the Discovery message on the domain represented by *dmm*, including the response from the addressed station.

The worst-case time span for a master *i* to enter into the logical ring, after a station *k* ($t_{master\_entry}^{k,i}$), can be calculated by Eq. (3.17). Nevertheless, this equation assumes that master *k* is on the Active_Idle state, but when master *k* detects that it is out of the original logical ring it goes into the Listen_Token state for two complete token rounds before being ready to enter the logical ring. Therefore, it is necessary to add 3 token cycles to the total time required for a master to enter the logical ring. For an illustration of the referred behaviour the reader is referred to Section 6.3.

It is worthwhile to point out that in order to reduce the time required for a master to enter the logical ring, the following should be accounted:

- the address of the mobile wireless masters must be as near as possible from the address of the fixed stations preceding them;
- preferably, the length of the GAP interval parameter should be 1, for every master in the domain, i.e. after every resident wireless master or BM, at most one mobile wireless master can enter into the ring;
- the *G* factor must be as small as possible.

Once the discovery of stations is complete, or a new master has entered into a different domain, the domain DMM sends a Route_Update message, which will be used by the bridges to update their routing tables. The worst-case time span that the

Route_Update message, relative to station *s*, needs to go from DMM *dmm* to a BM *bm* (this time span is denoted as $t_{RU,s}{}^{bm}$) can be calculated by $Ru_{RU,s}{}^{dmm \rightarrow bm}$ (using Eq. (7.6)).

To summarize, the time required before a BM *bm* knows that a station *s* is again operational in a wireless domain, the duration of Phase 4, is given by the following formulation:

$$t^{s,bm}_{phase4} = \begin{cases} t^{dmm}_{disc} + t^{bm}_{RU,s}, s \in \Pi_{slave} \\ t^{dmm}_{dis} + 3 \times t^{dmm}_{cycle} + t^{k,j}_{m\_master\_entry} + t^{bm}_{RU,s}, s \in \Pi_{master} \end{cases} \tag{7.16}$$

where, *dmm* represents DMM of the domain in which station *s* is, or to where it has entered. Π_{slave} and Π_{master} are the set of mobile wireless slaves and mobile wireless masters in the system, respectively.

7.3.5. Worst-Case IDMP duration

The worst-case duration of the inter-domain mobility procedure is measured from the sending of the Start_Mobility_Procedure message, by the GMM, until all mobile wireless stations are able to receive and make requests in normal operation. This time span can therefore be calculated by using the following equation:

$$t_{mob} = t_{phase1} + t_{phase2} + \max\left\{t^{dmm}_{phase3} + t^{s,dom}_{phase4}\right\}, \forall \; dmm \tag{7.17}$$

This quantity is only indicative about the performance of the system, since the effect of the inter-cell mobility procedure varies of as a function of the type of transactions, as it will be discussed later in Chapter 8.

7.3.6. Computing the Number of **Beacons**

The channel assessment method which is assumed in this thesis is based on the channel assessment method proposed for the repeater-based approach used in RFieldbus, as described in Chapter 2. In (Alves, 2003), the author had proposed a formulation which enables the calculation of the Beacon transmission duration, for the various Structuring Intermediate System, on a repeater-based network. In the remainder of this section, that work is adapted in order to obtain a formulation which enables the calculation of the number of Beacons to be transmitted by every wireless DMM ($n_{beacon}{}^{dmm}$).

The main problem of the channel assessment phase is related to the non synchronisation of the wireless domains in the overall network, which is due to the variability of the time required by the Start_Beacon_Transmission message to reach the system wireless DMMs. In Figure 7.8, it can be observed that the Beacon transmission sub-phase in the domains of BMs M5 and M7 start and end at different time instants. The consequence of this lack of synchronism is that the duration of the Beacon transmission sub-phase must be different for every domain. Additionally, its duration must guarantee that any mobile wireless station is capable of correctly assessing every wireless radio channel of the network and switch to the best channel available.

In (Alves, 2003), the author proposes a formulation which enables the calculation of the minimum time required by a mobile wireless station to correctly assess the quality of every wireless channel the assumed wireless network.

$$t_{ass} = (2 \times nch - 1) \times C_{bframe} + nch \times (t_{bgao} + t_{sw}) \tag{7.18}$$

nch is the number of channels in the network. C_{bframe} is the duration of the `Beacon` frame. t_{bgap} is the time interval between `Beacons`. t_{sw} is the time required by the radio circuitry to change to another channel.

The main difference between the repeater-based approach and the bridge-based approach is related to the delays encountered by the `Start_Beacon_ Transmission` message which might reach a DMM by its best-case latency (tbc_{SB}^{dmm}) or by its worst-case latency (t_{SB}^{dmm}). Therefore to compute the number of `Beacons` to be transmitted on every domain the following procedure is recommended:

1. compute the t_{SB}^{dmm} and tbc_{SB}^{dmm} for every wireless DMM in the network;
2. determine $max\{t_{SB}^{dmm}\}$, the DMM in the domain where: $max(t_{SB}^{dmm}) = t_{SB}^{dmm}$ is referred as *dmm_max*;
3. assume that *dmm_max* transmits the minimum number of `Beacons`, which can be calculated according to Eq. (7.18), and determine t_{ass_end}, which is equal to $max\{t_{SB}^{dmm}\} + t_{ass}$;
4. assuming that the `Start_Beacon_Transmission` message reaches the remaining wireless DMMs by its best time, determine $t'_{beacon\_tx}{}^{dmm}$, which is equal to $t_{ass\_end} - tbc_{SB}^{dmm}$;
5. finally, it is possible to compute the number of `Beacons` to be transmitted per domain computing $n_{beacons}^{dmm} = \lceil t'_{beacon\_tx}{}^{dmm} / C_{beacon} \rceil$.

7.4. Summary

This chapter starts by presenting a timing analysis of the worst-case response time of IDTs in the proposed bridge-based architecture, assuming that the mobility of stations between different wireless domains is not active. This timing analysis is based on the works for the single logical ring, which were presented in Chapter 3. These results are used on the rest of the chapter as the basis for a timing analysis related to the different phases of the IDMP, which will be used later in Chapter 8 in order to incorporate the effects of the IDMP on the worst-case response time analysis of IDTs.

Chapter 8

WCRT Analysis of Transactions Considering the Latencies of the IDMP

During the evolution of the Inter-Domain Mobility Procedure (IDMP), there are periods of time during which transactions between stations are disabled. The length of these periods depends on the type of stations involved in the transactions and the domains to which they belong. This chapter analyses the impact of the IDMP latencies and inaccessibilities on the Worst-case Response Time (WCRT) of Inter-Domain Transactions (IDT) and Intra-Domain Transactions (IADT).

8.1. Introduction

The IDMP requires a complex set of steps in order to ensure that its main objectives (no errors, no loss of messages and orderly delivery of messages) are meet. The assurance of these objectives is only possible at the cost of blocking the regular network activity during some parts of its progress. This occurs, for instance, after the reception of the `Start_Mobility_Procedure` message. Upon reception of this message, the system's BMs are unable to open new IDTs, to which they operate as BM_{ini}. Also, during the inquiry sub-phase and the `Beacon` transmission sub-phase, IADT are disabled.

The timing analysis presented for IDTs in Chapter 7 does not account for the delays referred above. These delays can have a significant impact on the WCRT, not only of IDTs but also of IADT.

In this chapter we will develop a detailed analysis of the impact of the IDMP latencies and inaccessibilities on the WCRT of IDT and IADT. This chapter is organized into two main sections. Section 8.2 analyses the different inaccessibility latencies caused by the IDMP. In Section 8.3 these inaccessibility periods are incorporated into the WCRT analysis of IADTs and IDTs.

8.2. Inaccessibility Periods due to the IDMP

The above mentioned inaccessibility periods are the following:

Case 1. the time during which IADT are disabled in a domain;

Case 2. time during which IDT are disabled in a Bridge Master (BM);

Case 3. time during which IDT addressed to mobile wireless stations are disabled on a BM;

Case 4. time during which mobile wireless stations are inaccessible.

Each of these cases will be addressed next in separate subsections.

8.2.1. Time During which IADT are Disabled in a Domain

During the IDMP there are periods of time in which IADTs are not allowed, specifically during the inquiry, the `Beacon` transmission and the station discovery sub-phases. In wired domains, IADT will be inhibited only during the inquiry sub-phase.

For the example network depicted in Figure 4.9, Figure 8.1 presents a timeline for part of the IDMP, which stresses in grey the duration of the periods during which IADT are disabled in domains D^1 and D^3 (the domains are represented by their respective DMMs). In the remaining of this chapter, all example scenarios are referred to the network configuration illustrated in Figure 4.9.

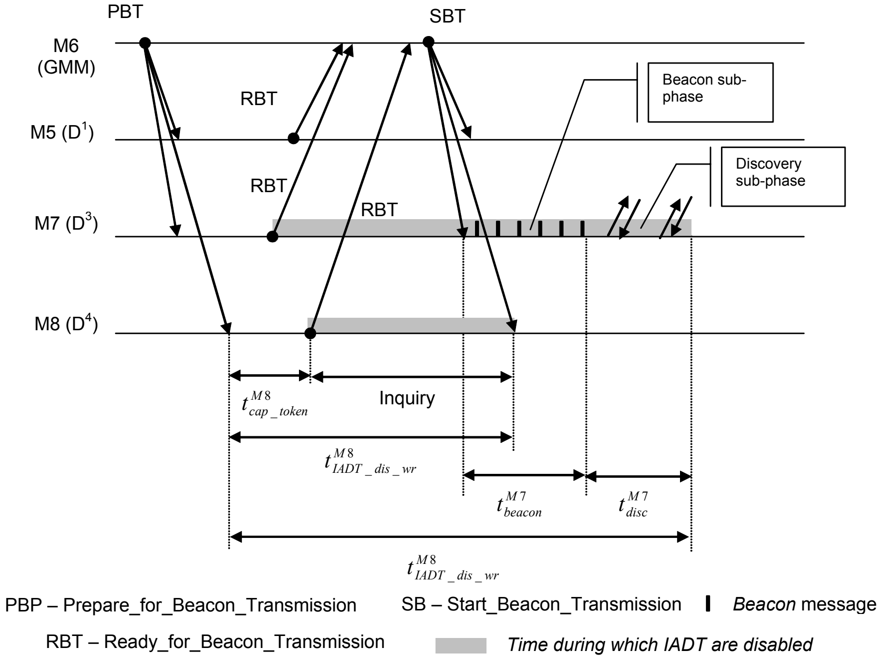


Figure 8.1 - Timeline representing the periods with IADT disabled

The time during which IADT are disabled in a domain starts after the capture of the token by the domain DMM, when entering into inquiry mode. This period ends after the station discovery sub-phase or after at the reception of the `Start_Beacon_Transmission` (SBT) message in wireless domains and in wired domains, respectively. Therefore, on a wired domain this time span is given by:

$$t^{dmm}_{IADT\_dis\_wr} = t_{phase2} - t^{dmm}_{PBT} - t^{dmm}_{cap\_token} + t^{dmm}_{SBT} \qquad (8.1)$$

The formulations for the calculation of the time spans t_{phase2}, $t_{PBT}{}^{dmm}$, $t_{cap\_token}{}^{dmm}$, $t_{SBT}{}^{dmm}$ and other in the remaining of this chapter were already provided in Chapter 7.

Note that wired terminating domains (like domain D^4) are not required to enter into the inquiry mode, and therefore, in such a domain, the time during which IADT are disabled is equal to 0.

For a wireless domain this time span is given by:

$$t^{dmm}_{IADT\_dis\_wl} = t_{phase2} - t^{dmm}_{PBP} - t^{dmm}_{cap\_token} + t^{dmm}_{phase3} + t_{disc} \qquad (8.2)$$

In both Eq. (8.1) and Eq. (8.2) *dmm* represents the domain.

8.2.2. Time During which IDT are Disabled in a BM

IDTs are disabled, on a BMs, from the reception of the `Start_Mobility_Procedure` (SMP) message until the end of the station discovery sub-phase or the reception of the `Start_Beacon_Transmission` (SBT) message, for wireless and wired domains, respectively. This scenario is illustrated in Figure 8.2. IDTs involving mobile wireless stations are only enabled again at the reception of a `Route_Update` (RU) message regarding the responder station. This scenario is analysed in Section 8.2.3.

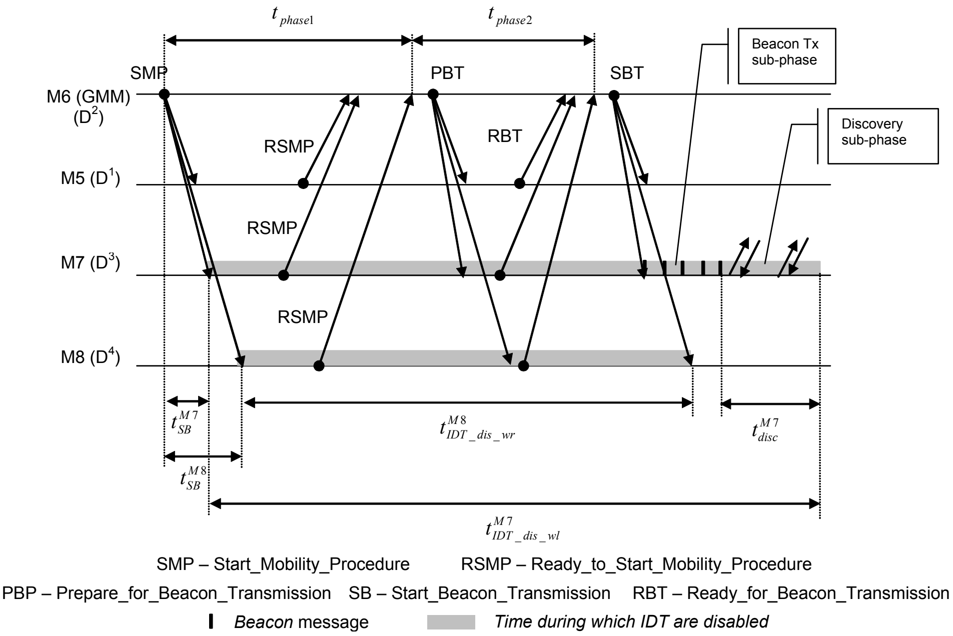


Figure 8.2 – Timeline representing the periods during which IDT are disabled

Figure 8.2, represents an scenario, which shows the progress of the IDMP on the network configuration depicted in Figure 4.9. The figure stresses in light grey the time span during which IDTs are disabled in BM M8 (${t_{IDT\_dis\_wr}}^{M8}$) and BM M7 (${t_{IDT\_dis\_wr}}^{M7}$).

It is important to note that after the end of the period of time during which IDT are disabled in a BM, it might be impossible to complete IDTs originating from stations in a domain D^x if an adjacent domain D^y still has its IDTs disabled. As illustrated in Figure 8.2, the wireless domain to which M7 belongs has its IDT disabled for a longer time than the domain of M8 (a wired domain). Nevertheless, an IDT request opens an IDT on its *BM_{ini} List of Open Transactions* (LOT), which can then be processed when the IDTs are enabled again on D^y.

The worst-case time span during which IDTs are disabled in a BM *bm*, in a wired domain, can be calculated by the following equation:

$$t^{bm}_{IDT\_dis\_wl} = t_{phase1} - t^{bm}_{SMP} + t_{phase2} + t^{dmm}_{SBT} \tag{8.3}$$

whereas for a wireless domain the worst-case time span is given by:

$$t^{bm}_{IDT\_dis\_wr} = t_{phase1} - t^{bm}_{SMP} + t_{phase2} + t^{dmm}_{phase3} + t^{dmm}_{disc} \tag{8.4}$$

8.2.3. Time During which IDT Addressed to Mobile Wireless Stations are Disabled

IDTs with mobile wireless stations are disabled, from the reception of the `Start_Mobility_Procedure` message until these stations enter a new domain (masters use the GAP Update mechanism and slaves the discovery mechanism) or are detected in the same domain (by the discovery mechanism).

The entrance of mobile wireless stations into a new domain triggers the broadcast of a `Route_Update` message by the domain DMM. When this message reaches a BM, it updates its routing tables and enables IDT with that specific mobile wireless station.

Figure 8.3 depicts an example which illustrates the case of mobile wireless master M3, when moving between domains D^1 and D^3. The figure shows, in light grey, the time span during which IDT addressed to M3 are disabled in BM M8.

As shown in Figure 8.3, the time needed by the `Route_Update` message to reach a BM is different between BMs, since that message uses different paths to reach each one of the BMs in the system. Additionally, mobile wireless stations can belong to different wireless domains. So, once again, this time is different between different BMs and depends on all possible combinations of original domain and destination domain.

The following equation allows the calculation of the worst-case time span during which IDTs related to a mobile wireless masters *s* are disabled in a BM *bm*:

$$t^{bm,s}_{IDT\_MM\_dis} = t_{phase1} + t_{phase2} + \max_{\forall wldmm}\left\{t^{wldmm}_{phase3} + t^{wldmm}_{dics} + t^{s,i}_{m\_master\_entry} + t^{bm}_{RU,s}\right\} - t^{bm}_{SMP} \tag{8.5}$$

In this equation *j* is the address of the station that will be the predecessor of station *s*, when *s* moves into a new domain. *wldmm* represents the set of DMMs which belong to all possible domains where station *s* is capable of entering. Note that, if a mobile wireless master moves between its original domain and the domain of BM *bm*, then $t^{bm}{}_{RU,s}$ is equal to zero, since BM *bm* detects the entrance of master *s* into the ring. For slaves, this time span is usually shorter, and it is given by:

$$t^{bm,s}_{IDT\_MS\_dis} = t_{phase1} + t_{phase2} + \max_{\forall wldmm}\left\{t^{wldmm}_{phase3} + t^{wldmm}_{disc} + t^{bm}_{RU,s}\right\} - t^{bm}_{SMP} \tag{8.6}$$

Also in this case, if the mobile wireless slave moves between its original domain and the domain of BM *bm*, then $t^{bm}_{RU,s}$ is equal to zero.

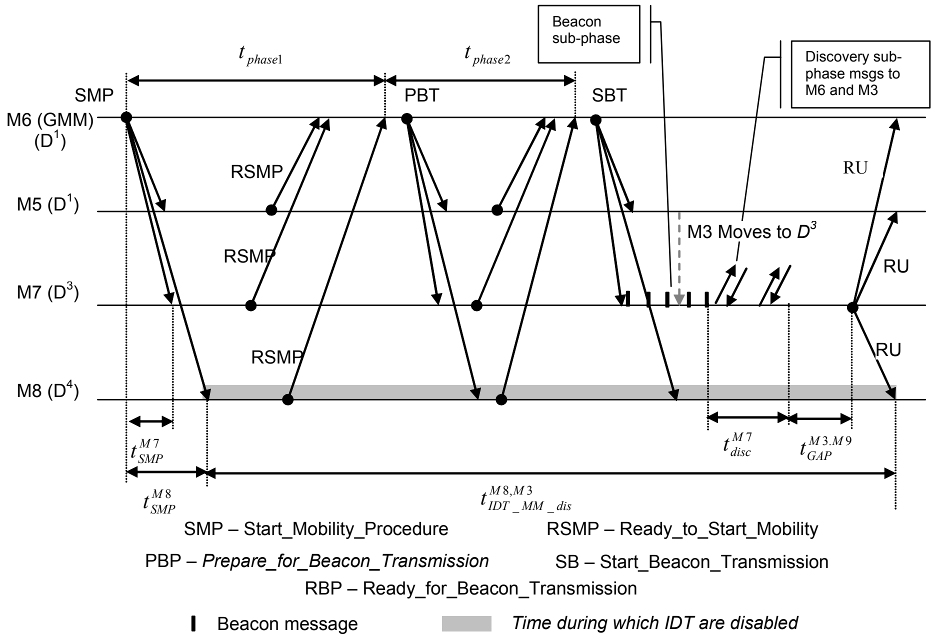

Beacon sub-phase
Discovery sub-phase msgs to M6 and M3
t_{phase1}
t_{phase2}
SMP
PBT
SBT
M6 (GMM) (D^1)
M5 (D^1)
M7 (D^3)
M8 (D^4)
RSMP
RSMP
RSMP
M3 Moves to D^3
RU
RU
RU
t^{M7}_{SMP}
t^{M8}_{SMP}
$t^{M8,M3}_{IDT\_MM\_dis}$
t^{M7}_{disc}
$t^{M3,M9}_{GAP}$
SMP – Start_Mobility_Procedure
RSMP – Ready_to_Start_Mobility
PBP – *Prepare_for_Beacon_Transmission*
SB – Start_Beacon_Transmission
RBP – Ready_for_Beacon_Transmission
Beacon message
Time during which IDT are disabled

Figure 8.3– Timeline illustrating the periods of time during which IDT are not possible with mobile wireless stations

8.2.4. Time During which Mobile Wireless Stations are Inaccessible

Mobile wireless stations are not capable of performing transactions, as initiators or responders, from the start of the inquiry sub-phase until a DMM detects its entrance into another domain or its continuation on the same domain. Figure 8.4 depicts a scenario were mobile wireless master M3 moves from domain D^1 to D^3. The time span during which M3 is inaccessible is represented in light grey.

If the original domain is represented by *dmm* and the destination domain by *dmm'*, then the following equation gives the worst-case time span during which a mobile wireless station *s* is inaccessible:

$$t_{ina}^{s,dmm\rightarrow dmm'} = t_{phase2} - t_{PBT}^{dmm} - t_{cap\_token}^{dmm} + t_{phase3}^{dmm'} + t_{phase4}^{s,dmm'} \tag{8.7}$$

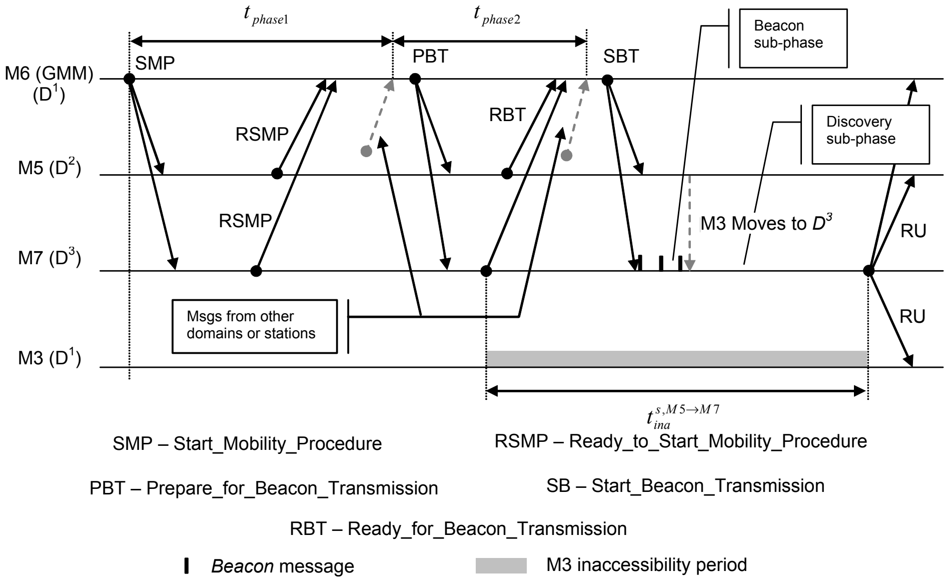

t_{phase1}
t_{phase2}
SMP
PBT
SBT
Beacon sub-phase
Discovery sub-phase
M6 (GMM) (D^1)
RSMP
RBT
M5 (D^2)
RSMP
M3 Moves to D^3
RU
M7 (D^3)
RU
Msgs from other domains or stations
M3 (D^1)
$t_{ina}^{s,M5\rightarrow M7}$
SMP – Start_Mobility_Procedure
RSMP – Ready_to_Start_Mobility_Procedure
PBT – Prepare_for_Beacon_Transmission
SB – Start_Beacon_Transmission
RBT – Ready_for_Beacon_Transmission
Beacon message
M3 inaccessibility period

Figure 8.4 – Timeline illustrating the period of time during which a mobile wireless master station is inaccessible

8.3. Incorporating the IDMP into the WCRT of Transactions

The WCRT analysis for a MLR PROFIBUS network, presented in Section 7.2, assumes that the IDMP is not active. We will now analyse the impact of the IDMP into:

- Intra-Domain Transactions (IADT);
- Inter-Domain Transactions (IDT) between wired or domain resident wireless stations belonging to different domains;
- Inter-Domain Transactions (IDT) involving mobile wireless stations as responders or initiators.

These will be addressed in detail in the following subsections.

8.3.1. Intra-Domain Transactions (IADT)

The period of time in which transactions between stations belonging to the same domain are not possible, comprises the inquiry sub-phase, the `Beacon` transmission sub-phase and the station discovery sub-phase. Eq. (3.1) and Eq. (3.4), which permit the calculation of the WCRT for IADTs, do not incorporate these delays. Therefore, it is necessary to update these equations by considering the period of time during which IADTs are disabled in a domain, which can be calculated using Eq. (8.1) or Eq. (8.2), for a wired and for a wireless domain, respectively.

The assumption is that a worst-case condition occurs when master k queues a request related to message stream S_i^k, just before the start of the period of time during which IADTs are disabled. The following equation incorporates these effects on the calculation of the WCRT for IADTs.

$$Rslr\_m_i^k = \begin{cases} Rslr_i^k + t_{IADT\_dis\_wr}^{dmm}, \text{in wired domains} \\ Rslr_i^k + t_{IADT\_dis\_wl}^{dmm}, \text{in wireless domains} \end{cases} \tag{8.8}$$

$t_{IADT\_dis\_wr}{}^{dmm}$ and $t_{IADT\_dis\_wl}{}^{dmm}$ are the time span during which IADTs are disabled in a wired and in a wireless domain, respectively.

The m in $Rslr\_m_i^k$ denotes that this equation incorporates the effects of the IDMP. In Eq. (8.8) the domain is defined by its respective DMM.

8.3.2. Inter-Domain Transactions (involving Domain Resident Wireless Stations or Wired Stations)

As explained in Section 8.2.2, only after $t_{IDT\_dis\_wr}{}^{dmm}$ and $t_{IDT\_dis\_wl}{}^{dmm}$, IDTs are enabled again, on wired and wireless domains, respectively. Additionally, IADTs are disabled during $t_{IADT_dis_wr}{}^{dmm}$or $t_{IADT\_dis\_wl}{}^{dmm}$ on wired and wireless domains, respectively. Therefore, any request related to an IDT that reaches a BM during these periods is affected by the IDMP. Consequently, the WCRT of an IDT depends on the relation between the message stream period, the duration of the period of time in which IDTs are disabled in a BM, and also on the duration of the period of time in which IADT are disabled in a domain.

In the case of an IDT related to message stream S_i^k, involving domain resident wireless stations or wired stations, the following situations are analysed separately:

Case 1. $T_i^k > t_{IADT\_dis}{}^{bm} > t_{IDT\_dis}{}^{bm}$
Case 2. $t_{IADT\_dis}{}^{bm} < T_i^k \leq t_{IDT\_dis}{}^{bm}$
Case 3. $T_i^k \leq t_{IADT\_dis}{}^{bm} < t_{IDT\_dis}{}^{bm}$

$t_{IADT\_dis}{}^{bm}$ represents either $t_{IADT\_dis\_wr}{}^{dmm}$ or $t_{IADT\_dis\_wl}{}^{dmm}$ on the initiator domain, in the case where the domain is wired or wireless, respectively. *bm* represents the BM, on master k domain, which is used as BM_{ini} by message stream S_i^k, and *dmm* is the DMM in the domain to which master k belongs. Note that $t_{IADT\_dis\_wl}{}^{dmm}$ is equal for every station on the *dmm*'s domain. $t_{IDT\_dis}{}^{bm}$ represents either $t_{IDT\_dis\_wr}{}^{bm}$ or $t_{IDT\_dis\_wl}{}^{bm}$, where *bm* represents the BM, on master k domain, which is used as BM_{ini} by message stream S_i^k.

Case One

In this case, the message stream period is larger than $t_{IADT\_dis}{}^{bm}$ and $t_{IDT\_dis}{}^{bm}$. Therefore, it is obvious to conclude that at most one request related to an IDT might be lost due to the IDMP. The following equation incorporates this case by adding another retry on the WCRT calculation for IDTs:

$$Rmlr\_m_i^k = \left(A_i^k + 1\right) \times T_i^k + Rslr_i^k \tag{8.9}$$

Case Two

Several requests related to message stream S_i^k can be lost during the evolution of the IDMP. Also, since $T_i^k \geq t_{IADT\_dis}^{bm}$, it is not possible to guarantee that a request is queued on the initiator transmission queue at some point in the period of inaccessibility during which IADT are disabled. Figure 8.5 depicts such kind of scenario, assuming the network scenario of Figure 4.9, and an IDT between master M2 and slave S7. In the depicted scenario the first and the second requests that arrive at M8/BM_{ini} are ignored. This is so, since M8/BM_{ini} had previously received a `Start_Mobility_Procedure` message, and stopped accepting new IDTs. M8/BM_{ini} only opens an IDT on the third request after the end of period of time during which IADTs are disabled in the initiator domain. The fourth request finishes the IDT.

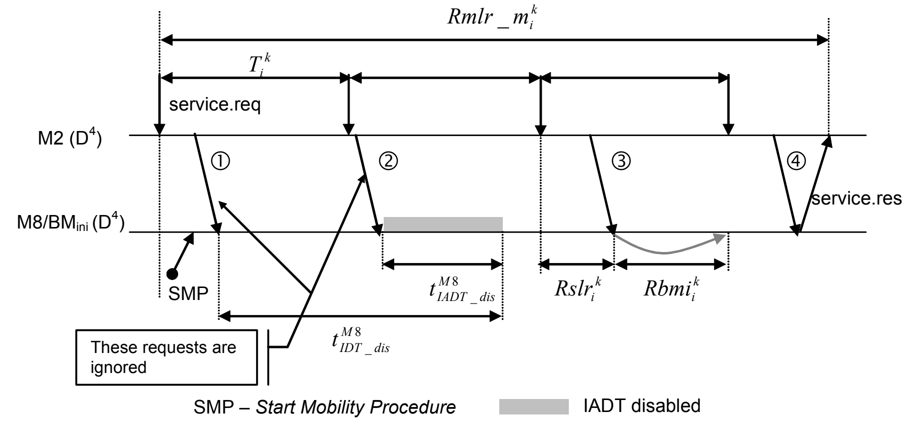


Figure 8.5 – Case when $t_{IADT\_dis}^{bm} < T_i^k \leq t_{IDT\_dis}^{bm}$

To incorporate this effect of IDMP into the IDT WCRT, when $t_{IADT\_dis}^{bm} < T_i^k \leq t_{IDT\_dis}^{bm}$, the following worst-case assumptions are made:

- the first request issued by master k, related to message stream S_i^k, arrives at BM_{ini} just after it had received the `Start_Mobility_Procedure` message;
- another request, which initialises an IDT on the BM_{ini} LOT, arrives after the end of the period of time during which IADTs are disabled in master k domain.

Under these conditions, the following equation accounts for the effect of the IDMP on IDTs if $t_{IADT\_dis}^{bm} < T_i^k \leq t_{IDT\_dis}^{bm}$:

$$Rmlr\_m_i^k = \begin{cases} Rslr_i^k + t_{IDT\_dis\_wr}^{bm} + T_i^k + Rmlr_i^k & \text{in wired domains} \\ Rslr_i^k + t_{IDT\_dis\_wl}^{bm} + T_i^k + Rmlr_i^k & \text{in wireless domains} \end{cases} \tag{8.10}$$

Case Three

In this case, since $T_i^k < t_{IADT\_dis}^{bm}$, master k is able to queue at least one retry related to message stream S_i^k, at some point in the period of inaccessibility during which IADTs are disabled. Figure 8.6 depicts such kind of scenario.

In this specific example, the first two requests are ignored, since M8/BM_{ini} has previously received the `Start_Mobility_Procedure` message. The third request is queued on the M2 output queue at some stage in the period of time during which IADTs are disabled. As soon as this period ends, and M2 is able to compete for the medium, the request is transmitted, initialising an IDT in M8/BM_{ini}. The fourth request is ignored by M8/BM_{ini} since it does not have any response available. Finally, only on the fifth request a response is transmitted back to M2.

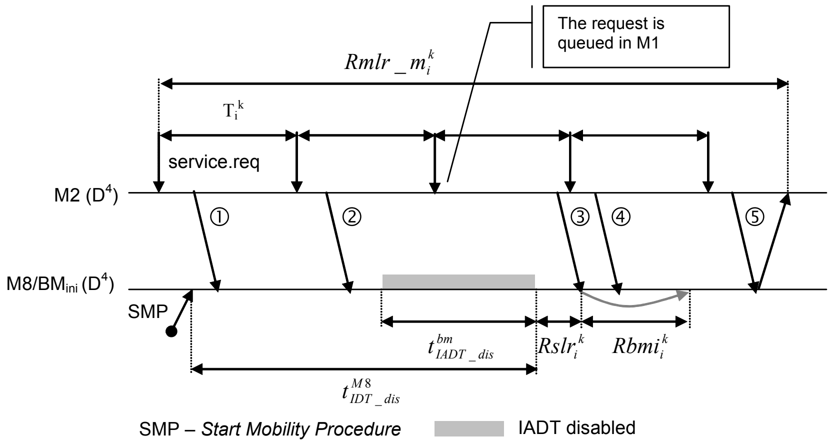


Figure 8.6 – Case when $T_i^k \le t_{IADT\_dis}{}^{bm} < t_{IDT\_dis}{}^{bm}$

To obtain the effect of the IDMP on the IDT response time, when $T_i^k \le t_{IADT\_dis}{}^{bm} < t_{IDT\_dis}{}^{bm}$, the following assumptions are made:

- the first request issued by master *k* related to message stream S_i^k, arrives at BM_{ini} just after it had received the `Start_Mobility_Procedure` message;
- another request related to the same stream is received by BM_{ini} at some stage in the period of time during which IADTs are disabled.

If the conditions exposed above the hold, the following equation accounts for the IDMP effects on the WCRT related to IDTs:

$$Rmlr\_m_i^k = \begin{cases} Rslr_i^k + t_{IDT\_wr\_dis}^{bm} + Rmlr_i^k & \text{, in wired domains} \\ Rslr_i^k + t_{IDT\_wl\_dis}^{bm} + Rmlr_i^k & \text{, in wireless domains} \end{cases} \tag{8.11}$$

Accounting for the effects of the IDMP on the calculation of $Rbmi_i^k$

The cases presented above in this Section 8.3.2 only take into consideration the state in the initiator domain. Nonetheless, there are no guarantees about the state of the remaining BMs which belong to the IDT Communication Path. In fact, in wireless domains IDTs are disabled for a longer time, due to the transmission of `Beacons` and the station discovery sub-phase. In such cases, when an IDF arrives at a bridge, having one of its BMs with IDTs disabled, the IDF must wait on the BM output queue until being able to be relayed by the BM.

Figure 8.7 depicts an example regarding an IDT between master M2 and slave S1. In this example, the two first requests transmitted by M2 are ignored since M8/BM_{ini} has its IDTs disabled; the third request is accepted by M8/BM_{ini} opening an IDT at M8/BM_{ini} LOT. M8/BM_{ini} receives the request and transforms it into an IDF using the rules defined by the IDP, and then this frame is relayed until reaching M6, which does not have its IDTs enabled. Consequently, the IDF has to wait (t_{delay}^{M6}) until the IDTs are again enabled in M6.

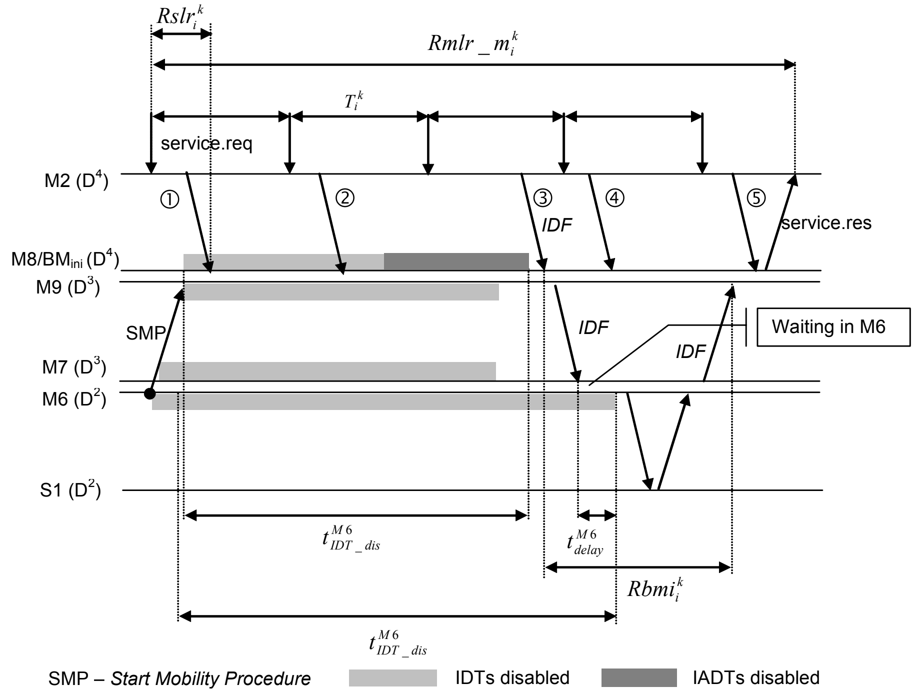


Figure 8.7 – Delays due to the BMs in the communication path set not having its IDTs enabled

The following equation provides a new formulation to $Rbmi_i^k$ which includes the possible effects of the IDMP for all the cases described in this section:

$$Rbmi\_m_i^k = Rbmi_i^k + \max_{\forall bm \in \Omega_{req}(S_i^k,t)} \left\{ pos(t_{IDT\_dis\_GMM}^{bm} - t_{IDT\_dis\_GMM}^{BM_{ini}}) \right\} \qquad (8.12)$$

$t_{IDT\_dis\_GMM}^{BM_{ini}}$ is the time span during which IDTs are disabled on the BM which is the BM_{ini} for message stream S_i^k, added to the time required for the `Start_Mobility_Procedure` message to reach that BM. Since the evolution of the IDMP is asynchronous, this time span is referenced to the start of the IDMP by the GMM.

Therefore, $t_{IDT\_dis\_GMM}^{BM_{ini}}$ is equal to $t_{IDT\_dis}^{BM_{ini}}$+ $t_{SMP}^{BM_{ini}}$. Similarly, $t_{IDT\_dis\_GMM}^{bm}$ refers to the time during which IDTs are disabled on a BM *bm* which belongs to the IDTreq Communication Path related to message stream S_i^k ($\Omega_{req}(S_i^k, t)$), added to the time required for the `Start_Mobility_Procedure` message to reach BM *bm*. Therefore, $t_{IDT\_BM}^{bm}$ can be calculated by $t_{IDT\_dis}^{bm}$ + t_{SMP}^{bm}. In the calculation of $t_{IDT\_dis\_GMM}^{BM_{ini}}$ and $t_{IDT\_dis\_GMM}^{bm}$, the inclusion of the time spans $t_{SMP}^{BM_{ini}}$ and t_{SMP}^{bm} is required in order to have the same time reference – the start of the IDMP by the system GMM. In Eq. (8.12), *pos*(*a*) is a function that returns *a* when $a \geq 0$ and 0 otherwise

8.3.3. Transactions Involving Mobile Wireless Stations

The problem of providing a worst-case bound for the response time of IDTs related to a message stream S_i^k involving mobile wireless stations is, in practice, similar to the scenario described in subsection 8.3.2.

Three main cases must be considered:

Case 1. IDTs between a wired or domain resident wireless master and a mobile wireless slave/master;

Case 2. IDTs between a mobile wireless master and a wired or domain resident wireless slave/master;

Case 3. IDTs involving two mobile wireless stations.

Case One

In this case, IDTs involving a mobile wireless station are disabled, on the BM_{ini}, from the reception of the `Start_Mobility_Procedure` message until the reception of a `Route_Update` message regarding the responder station *s* – $t_{IDT\_mob\_dis}^{BM_{ini},s}$. This time span can be calculated using Eq. (8.5) or Eq. (8.6), in the case where the responder is a master or the responder is a slave, respectively.

Figure 8.8 depicts an example which illustrates an IDT between a master M2 and a mobile wireless slave S6, also for the scenario illustrated in Figure 4.9. In this example, the requests transmitted by M2 and received by M8/BM_{ini} are ignored from the reception of the `Start_Mobility_Procedure` message until the reception of a `Route_Update` message regarding slave S6. The third retry is successful, and opens an IDT transaction on M8/BM_{ini} LOT.

To obtain the worst-case response time for an IDT regarding a message stream S_i^k, the following worst-case conditions are assumed:

- the first request is received by BM_{ini} just after the reception of the `Start_Mobility_Procedure` message;
- no IDT request are accepted by BM_{ini} during $t_{IDT\_mob\_dis}^{BM_{ini},s}$, i.e. from the reception of the `Start_Mobility_Procedure` message until the reception of a `Route_Update` message regarding station *s*;
- a request arrives at BM_{ini} just before it receives the `Route_Update` message concerning the responder station.

The following formulation gives the WCRT for IDTs made between a wired or domain resident wireless master *k* and a mobile wireless station *s*:

$$Rmlr\_m_i^k = Rslr_i^k + t_{IDT\_mob\_dis}^{BM_{ini},s} + T_i^k + Rmlr_i^k \tag{8.13}$$

The time span $t_{IDT\_mob\_dis}{}^{BM_{ini},s}$ can be calculated by Eq. (8.5) or (8.6), in the case when the mobile wireless station is a master or a slave, respectively. It is important to note that, due to the definition of Eq. (8.5) and (8.6), Eq. (8.13) already accounts for the all the possible locations of the mobile wireless station *s*.

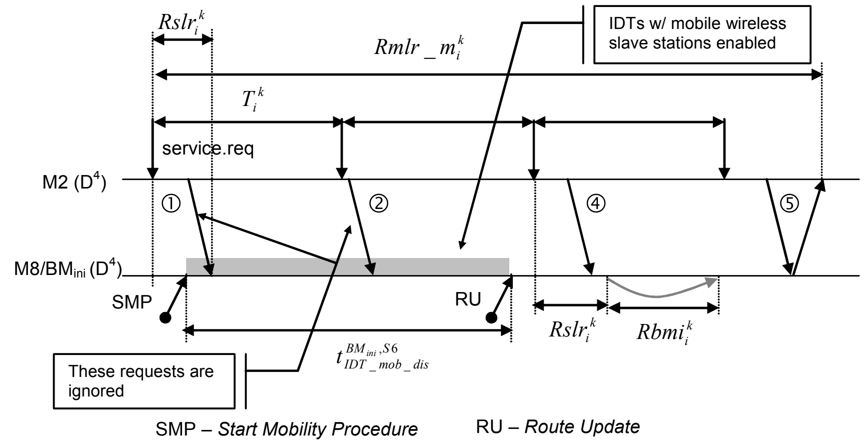

$Rslr_i^k$
$Rmlr\_m_i^k$
IDTs w/ mobile wireless slave stations enabled
T_i^k
service.req
M2 (D^4)
①
②
④
⑤
M8/BM$_{ini}$ (D^4)
SMP
RU
$Rslr_i^k$
$Rbmi_i^k$
$t_{IDT\_mob\_dis}^{BM_{ini},S6}$
These requests are ignored
SMP – *Start Mobility Procedure*
RU – *Route Update*

Figure 8.8 – Example IDT with a mobile wireless station

Case Two

In this case, the IDT initiator is a mobile wireless master, and the responder is a wired or domain resident wireless station.

Figure 8.9 shows a timeline depicting an IDT between master M3 and slave S1, where master M3 moves, using the IDMP, from the original domain (D^1) to a destination domain (D^3), during the execution of an IDT with S1.

As it is illustrated in Figure 8.9, after the reception of the Start_Mobility_Procedure message by the IDT $(BM_{ini})^{orig}$, on the original domain (the domain D^1, to which M5/$(BM_{ini})^{D1}$ belongs to), M3 is no longer able to complete IDTs with S1. Only after entering the logical ring on the destination domain M3 is capable of completing the transaction, using BM M7/$(BM_{ini})^{D3}$ on the destination domain.

To obtain the worst-case response time for an IDT related to message stream S_i^k, the following worst-case conditions are assumed:

- the first request related to S_i^k is received by $(BM_{ini})^{orig}$ just after the reception of the Start_Mobility_Procedure message;
- the first request related to S_i^k, made on the destination domain, is delayed by T_i^k after master *k* has entered the domain.

In these conditions, the following equation updates Eq. (7.1) for the case when the transaction is made between a mobile wireless master and a wired or domain resident wireless station:

$$Rmlr\_m_i^k = (Rslr_i^k)^{orig} + (t_{IDT\_MMM\_dis}^k)^{orig \to dest} + T_i^k + (Rmlr_i^k)^{dest} \tag{8.14}$$

In this equation the expression $(y)^d$ represents the value for timing y in domain d, e.g. $(Rslr_i^k)^{orig}$ represents the WCRT for message stream i form master k in the original domain of master k. $(t_{IDT\_MMM\_dis}{}^k)^{orig \to dest}$ represents the worst-case time during which master k has its IDTs disabled when moving between the original domain and the destination domain, which can be calculated as follows:

$$(t_{IDT\_MMM\_dis}^k)^{orig \to dest} = t_{phase1} + t_{phase2} + t_{phase3}^{dest} + (t_{m\_master\_entry}^{k,j})^{dest} - t_{SMP}^{(BM_{ini})^{orig}} \tag{8.15}$$

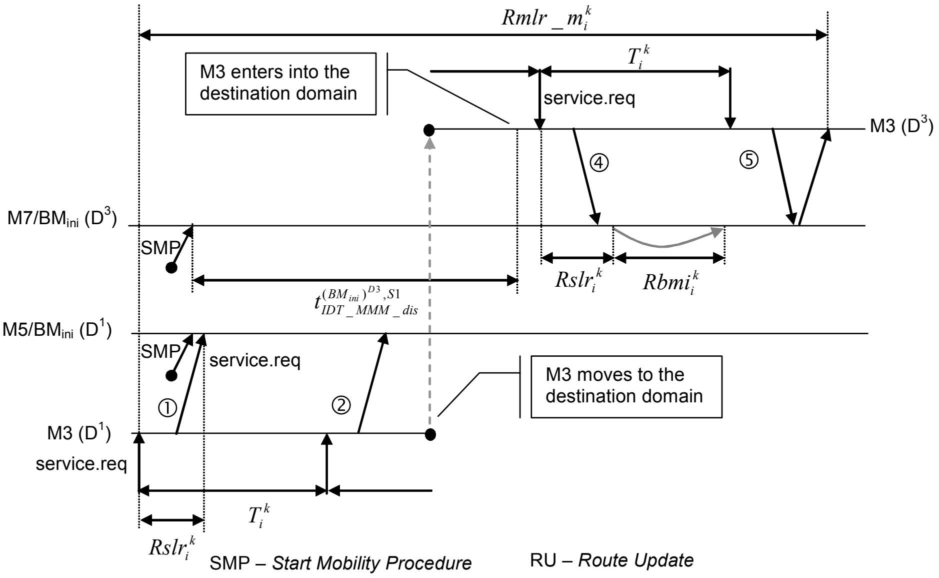


Figure 8.9 – Example IDT between mobile wireless master and slave S1

Case Three

Finally, the third case occurs when the two stations move during the execution of an IDT. In this case, it is also necessary to consider two sub-cases:

- the `Route_Update` message, regarding the responder station, arrives at the destination domain **before** the initiator has entered into the domain;
- the `Route_Update` message, regarding the responder station, arrives at the destination domain **after** the initiator has entered into the domain.

To distinguish between the two sub-cases, the following equation allows determining the time when the initiator is operational (i.e. when it is capable of making transactions) on its destination domain (*dest_i*):

$$(t_{ini\_op})^{dest_i} = t_{phase1} + t_{phase2} + t_{phase3}^{dest_i} + (t_{disc})^{dest_i} + (t_{m\_master\_entry}^{k,j})^{dest_i}$$

and when the `Route_Update` message regarding the entry of the responder (r) in its destination domain ($dest_r$) has reached the BM_{ini} on the destination domain of the initiator:

$$(t_{resp\_op})^{dest_r} = t_{phase1} + t_{phase2} + t_{phase3}^{dest_r} + t_{phase4}^{r,BM_{ini}}$$

In the first sub-case, the `Route_Update` message arrives at the destination domain BM_{ini} before master k is ready to make the request. Therefore, the conditions are similar to case two (i.e. when the IDT responder is a domain resident master/slave) and the WCRT can be given by Eq. (8.14).

In the second sub-case, the initiator has to wait until the reception of a `Route_Update` message before being capable of completing a transaction with the responder. Therefore, the WCRT can be calculated based partially on Eq. (8.13), in order to account for the mobility of the initiator station as follows:

$$\begin{aligned} Rmlr\_m_i^k = {} & (Rslr_i^k)^{orig} + t_{phase1} + t_{phase2} + t_{phase3}^{dest_i} + (t_{disc})^{dest_i} + (t_{m\_master\_entry}^{k,j})^{dest_i} + \\ & - t_{SMP}^{orig} + T_i^k + (Rmlr_i^k)^{dest} \end{aligned} \tag{8.16}$$

8.4. Summary

This chapter analysed the impact of the IDMP on the response time of IADT and IDT on the proposed hybrid wired/wireless PROFIBUS bridge-based network. The impact translates on additional latencies for the message streams. These latencies depend on the type of stations involved, the type of transactions and their mobility pattern.

The analytical model developed in this section is used in the Chapter 9 to illustrate its applicability in order to determine the timing behaviour of the proposed bridge-based network on a specific scenario.

Chapter 9

Numerical Examples and Performance Comparisons

This chapter presents a numerical example showing how the worst-case timing analysis presented in Chapters 7 and 8 can be applied. It also presents the results extracted from a simulation tool of the proposed hybrid wired/wireless bridge-based network. The chapter finalises by presenting a performance comparison of the proposed architecture with the repeater-based approach (RFieldbus), based on the worst-case timing analyses presented for the two types of networks.

9.1. Introduction

The main objectives of this chapter are to exemplify how the worst-case timing analysis and models presented in Chapters 4, 7 and 8, related to the Inter-Domain Protocol (IDP) and to the Inter-Domain Mobility Procedure (IDMP), can be applied to a specific network scenario.

The timing analysis proposed in this thesis always assumes worst-case scenarios. Therefore a simulator tool was developed, which enables emulating the behaviour of the network and to obtain some results regarding its operation on a controlled environment. The obtained results validate the proposed timing analysis and also provide statistical results regarding the network operation.

This chapter is organised as follows. Based on the network model proposed in Chapter 4, Section 9.2 presents the model of an example network scenario, including all the aspects regarding the diverse parameters of the model. Section 9.3 describes in detail how to obtain the WCRT related to IDTs and IADTs, without considering the influence of the IDMP, and thus using the methodologies presented in Chapter 7. Section 9.4 illustrates how to obtain the latencies of the different phases of the IDMP, which are then incorporated into the WCRT related to IADTs and IDTs considering the effects of mobility (Section 9.5). Section 9.6 compares the timing analysis results with results obtained by simulation, and Section 9.7 compares the timing behaviour of the proposed bridge-based approach with the repeated-based approach.

9.2. Network Example

The network topology presented in Figure 9.1 will be used throughout this chapter to illustrate the application of the results provided so far in this thesis.

The example network comprises two structured wireless domains D^1 and D^3, and two wired domains D^2 and D^4. There are two wired masters {M1 and M2}, one mobile wireless master {M3}, five wired slaves {S1, S2, S3, S4, S5}, one domain resident wireless slave {S7} and one mobile wireless slave {S6}.

BM M6 is the GMM and, at the same time, the DMM for domain D^2. M5, M7 and M8 are the DMMs for domains D^1, D^3 and D^4, respectively. BM M7 and BM M5 also have Base Station functionality, thus structuring wireless domains D^1 and D^3, respectively.

It is assumed that the Physical Layer (PhL) of the wireless stations is similar to the one developed for the RFieldbus project, which is based on 802.11b DSSS operating at 2 Mbit/s. The PhL of the wired domains is a standard PROFIBUS one, operating based on RS-485.

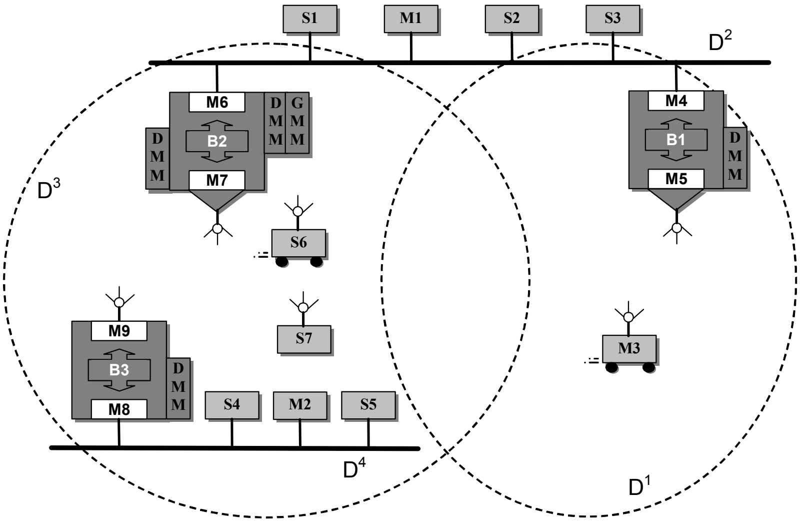


Figure 9.1 – Bridge-based Multiple Logical Ring (MLR) network example

The network is defined (Eq. (4.1)) by its set of Domains, Masters, Slaves and Intermediate Systems (IS):

$$N = \left(\left\{D^1, D^2, D^3, D^4\right\}, \left\{M1, M2, ..., M9\right\}, \left\{S1, S2, ..., S6\right\}, \left\{B1, B2, B3\right\}\right) \tag{9.1}$$

Each domain is characterised by its parameters, type of domain, physical medium, associated ISs, masters and slaves, and finally by the PROFIBUS specific parameters. Table 9.1 depicts the domain PROFIBUS parameters.

Table 9.1 – Domain PROFIBUS parameters

Domain	Parameters
D^1	(SWlD, ω^1, B1, {M5,M3}, -, D_PPAR^1)
D^2	(WrD, ω^2, {B1, B2}, {M1, M4, M6}, -, D_PPAR^2)
D^3	(SWlD, ω^1, {B2,B3}, {M7,M9}, S6, D_PPAR^3)
D^4	(WrD, ω^3, B3, M8, {M2, S4, S5}, D_PPAR^4)

Table 9.2 depicts the PROFIBUS specific parameters (Slot Time (T_{SL}), Gap Update factor (G), Highest Station Address (HSA), maximum number of DLL retries and Target Token Rotation Time (T_{TR})), which are defined by Eq. (4.3). These values are common to all stations in a specific domain.

Table 9.2 – Domain PROFIBUS parameters

Domain	Parameters
D_PPAR^1	(115, 1, 5, 1, 300)
D_PPAR^2	(115, 100, 6, 1, 300)
D_PPAR^3	(115, 1, 9, 1, 300)
D_PPAR^4	(115, 100, 8, 1, 300)

The T_{SL} parameter was set according to the recommendation of the PROFIBUS standard (EN50170, 1996), specifically of its Part 4.2 - Data Link Layer Protocol Specification.

The HSA is set differently for each domain according to the highest address for all stations belonging to that domain. This setting reduces the impact of the GAP Update mechanism, since in this way the master with the highest address has a minimum number of station addresses to inquiry (from 0 to the station with the lowest address in the logical ring). For details on the GAP Update mechanism, the reader is referred to Chapter 2.

Another important detail concerns the Gap Update factor (G), which is set to 1 in the wireless domains, in order to have the GAP Update mechanism always active. This feature effectively increases the network load, but since the FDL_Request_ Status frames used by the GAP Update mechanism have low-priority, the response time of high-priority message streams does not increase. In the wired domains, where the dynamic entrance of new stations is not expected, the Gap Update factor is set according to the PROFIBUS-DP standard, which recommends its setting to 100. The T_{TR} has been considered according to the formulation proposed in (Tovar and Vasques, 1999a). Therefore, the T_{TR} has been set to 300 bit times, which allows the transmission of one high or one low-priority message per token visit (if the token is not in delay).

Each domain is also characterised by its physical medium parameters (Eq. (4.4)): bit rate, head length, tail length and number of bits per DLL character. Table 9.3 presents the parameters for the domains in Table 9.1. Its settings reflect the capabilities of the system to accommodate domains with different data rates and different frame's formats, in the same network.

It is assumed that the wireless domains, D^1 and D^3, are using the 802.11b DSSS PhL at 2.0 Mbit/s, coding every character using 8 bits. The frames have a head of 32 bits and no tail. The reasons for the use of a frame head are related to the specific requirements of the DSSS modulation schema used by 802.11b. These bits are used by the receiver to acquire the incoming signal and synchronise the demodulator.

The wired domains, D^2 and D^4, use a standard PROFIBUS PhL operating at 1.5 Mbit/s and 500 kbit/s, for domains D^2 and D^4, respectively. Since these domains use the RS-485 standard for the transmission of the PhL frames, each character is coded using 11 bits. The three additional bits are related to one start, one stop, and one parity check bit. In wired domains, the PhL frames do not have a head or a tail sequence of bits.

Table 9.3 – Physical media parameters

Physical medium	Parameters
ω^1	(2000000, 32, 0, 8)
ω^2	(1500000, 0, 0, 11)
ω^3	(500000, 0 , 0, 11)

Master and slave stations are characterised by their respective parameters (Eq. (4.6) and (4.8), respectively) as presented in Table 9.4 . In the case of a master by the station type, mobility functionalities, set of message streams, number of high and low-priority message streams and the PROFIBUS master specific parameters. A slave is characterised by the station type, address, mobility functionalities and PROFIBUS slave specific parameters.

Table 9.4 – Master and slave parameters

Master	Parameters	Slave	Parameters
M1	(WrM, NONE, St^{M1}, 3, 0, M_PPAR^1)	S1	(WrS, NONE, S_PPAR^1)
M2	(WrM, NONE, St^{M2}, 3, 0, M_PPAR^2)	S2	(WrS, NONE, S_PPAR^2)
M3	(MWlM, MS, St^{M3}, 2, 0, M_PPAR^3)	S3	(WrS, NONE, S_PPAR^3)
M4	(WlBM, BR, -, -, -, M_PPAR^4)	S4	(WrS, NONE, S_PPAR^4)
M5	(WlSBM, DMM, -, -, -, M_PPAR^5)	S5	(WrS, NONE, S_PPAR^5)
M6	(WrBM, GMM, -, -, -, M_PPAR^6)	S6	(MWlS, MS, S_PPAR^6)
M7	(WlSBM, DMM, -, -, -, M_PPAR^7)	S7	(RWlS, NONE, S_PPAR^7)
M8	(WrBM, DMM, -, -, -, M_PPAR^8)		
M9	(WlBM, BR, -, -, -, M_PPAR^9)		

Each master also requires the definition of its PROFIBUS specific parameters: MAC address, T_{rdy}, T_{sdi}, $min(T_{sdr})$, $max(T_{sdr})$, T_{set} and T_{qui}, defined by Eq. (4.7). The instantiation of these, for the proposed network example is provided in Table 9.5. These parameters were chosen based on typical values proposed on the PROFIBUS-DP specification. Nevertheless, since this standard does not define any values for 2.0 Mbit/s bit rates, then it is assumed that the parameters recommended for 1.5 Mbit/s are still valid at a bit rate of 2.0 Mbit/s.

Domain D^4 is operating at a lower data rate, thus its PROFIBUS parameters are set according to the PROFIBUS-DP recommendations for a 500 kbit/s bit rate.

Table 9.5 – Master PROFIBUS parameters

Domain	Parameters
M_PPAR^1	(1, 10, 100, 11, 100, 1, 0)
M_PPAR^2	(2, 10, 90, 11, 70, 1, 0)
M_PPAR^3	(3, 10, 100, 11, 100, 1, 0)
M_PPAR^4	(4, 10, 100, 11, 100, 1, 0)
M_PPAR^5	(5, 10, 100, 11, 100, 1, 0)
M_PPAR^6	(6, 10, 100, 11, 100, 1, 0)
M_PPAR^7	(7, 10, 100, 11, 100, 1, 0)
M_PPAR^8	(8, 10, 90, 11, 70, 1, 0)
M_PPAR^9	(9, 10, 100, 11, 100, 1, 0)

According to Eq. (4.9), PROFIBUS slave parameters are only its address and $minT_{SDR}$ (Table 9.6), with the latter being set according to the PROFIBUS-DP recommendation.

Table 9.6 – Slave PROFIBUS parameters

Domain	Parameters
S_PPAR^1	(32, 11)
S_PPAR^2	(33, 11)
S_PPAR^3	(34, 11)
S_PPAR^4	(35, 11)
S_PPAR^5	(36, 11)
S_PPAR^6	(37, 11)
M_PPAR^7	(38, 11)

Each bridge IS is defined by the following parameters according to Eq. (4.10): IS type, internal relaying delay, internal forwarding delay, set of BMs and by the number of `Beacons` to be transmitted. Table 9.7 provides the values for these parameters for the network example. Note that bridge B3 does not transmit any `Beacons`, since it does not contain the base station functionality in its BM M9 and M8, the DMM of domain D^4 is connected to a wired domain.

Table 9.7 – ISs parameters

IS	Parameters
B1	(SLIS, 0, 0.03, {M4, M5}, 12)
B2	(SLIS, 0, 0.03, {M6, M7}, 13)
B3	(SLIS, 0, 0.03, {M8, M9}, 0)

Masters M1, M2 and M3 have the set of message streams as presented in Table 9.8. Each message stream (previously defined by Eq. (4.11)) contains the transaction responder, the size of the request and the response, the message stream period, the set of BMs on the IDreq communication path, the set of BMs on the IDres communication path and the priority of the stream.

The set of message streams presented in Table 9.8 tries to illustrate some probable transaction scenarios the network. Therefore, streams S_1^{M1} and S_2^{M1} are IADTs, between master M1 and slaves S1 and S2, respectively. The third message stream is an IDT between M1 and S5. M2 belongs to domain D^4 and is responsible for two message streams with stations in other domains. S_1^{M2} is an IDT involving two wired stations. S_2^{M2} is also an IDT, but between M2 and mobile wireless slave S6. M3 illustrates the case of message streams triggered in a mobile wireless master. S_1^{M3} is an IDT with wired slave S4, which belongs to domain D^4. S_2^{M3} is an IDT between two mobile wireless stations. Finally, S_3^{M3} is an IDT between a mobile wireless master and wired slave S3. In the following table, it is assumed that the location of the mobile wireless station is as depicted in Figure 9.1.

Table 9.8 – Message Streams

Stream	Parameters	Stream	Parameters
S_1^{M1}	(S1, 15, 20, 5, -, -, high)	S_2^{M2} (, D^3)	(S6, 15, 20, 5, {M9}, {M8}, high)
S_2^{M1}	(S2, 15, 20, 5, -, -, high)	S_1^{M3} (D^1)	(S4, 15, 20, 5, {M4, M7, M8}, {M5, M6, M9}, high)
S_3^{M1}	(S5, 15, 20, 5, {M7, M8}, {M6, M9}, high)	S_2^{M3}(D^1, D^3)	(S6, 15, 20, 5, {M4, M7}, {M5, M6}, high)
S_1^{M2}	(S3, 15, 20, 5, {M9, M6}, {M8, M7}, high)	S_3^{M3}(D^1)	(S3, 15, 20, 5, {M4}, {M5}, high)

9.3. Computing the Duration of Message Transactions

The transmission time of request and response frames related to a message stream depends on the parameters of the domains and stations involved, and also on the overhead caused by the IDP.

Since all message streams have been defined as having the same size, Table 9.9 presents the message cycle durations on several situations. In case the initiator and the responder belong to the same domain, the IADT acronym is used. The other table entries are related to the time required to transmit a request (denoted as IDReq) or a response (denoted as IDRes), both coded using the IDP frame coding.

The duration of the frames (or transactions) is computed using Eq. (4.11) and Eq. (4.12). In these results, we are assuming no DLL retries.

Table 9.9 – Message transaction duration

Type	Location	Duration (ms)
IADT	$D^1 = D^3$	0.265
	D^2	0.390
	D^4	1.110
IDReq	$D^1 = D^3$	0.072
	D^2	0.110
	D^4	0.330
IDRes	$D^1 = D^3$	0.092
	D^2	0.147
	D^4	0.440

9.3.1. Obtaining the Number of Message Streams Relayed by the BMs

To obtain the number of high-priority message streams relayed by the BMs, it is necessary to determine if a message stream is relayed by a BM or not, as depicted in Table 9.10. After that, the number of high-priority message streams can be determined for every BM as the sum of every individual message stream relayed by the BM. Note that, if a message stream is relayed by the same BM on different network configurations, then it should be accounted as a single entry. As an example, in Table 9.10, message stream S_2^{M2} (between M2 and S6) is relayed by BM M9 both in the case when S6 belongs to D^1 and in the case when S6 belongs to D^3. This kind of cases is denoted in Table 9.10 by the double-lined rectangle.

Table 9.10 – Number of Message Stream relayed by a BM

	S_3^{M1}	S_1^{M2}	$S_2^{M2}(D^1)$	$S_2^{M2}(D^3)$	$S_1^{M3}(D^1)$	$S_1^{M3}(D^3)$	S_2^{M3} (D^1,D^3)	S_2^{M3} (D^3,D^1)	$S_3^{M3}(D^1)$	$S_3^{M3}(D^3)$	*nh*
M4	-	-	1	-	1	-	1	1	1	-	4
M5	-	-	1	-	-	-	-	1	-	-	2
M6	-	-	1	-	1	-	1	1	-	1	4
M7	1	1	1	-	1	-	1	-	-	-	5
M8	1	-	-	-	1	1	-	-	-	-	2
M9	1	1	1	1	1	-	-	-	-	-	4

Obviously, these assumptions are very pessimistic since it is considered that almost all station location possibilities can occur at the same instant. Nevertheless, these assumptions help in reducing the complexity of the timing analysis.

It is important to note that the number of possible network configurations can be very high. It depends on the number of mobile wireless stations in the network (*n_mst*), and on the number of possible domains into which these stations can enter (*nd*). The following equation allows the calculation of the total number of possible network configurations (*n_conf*):

$$n\_conf = nd^{n\_mst} \tag{9.2}$$

Just as an example, in the network scenario under consideration, 4 different network configuration cases are possible. In the case of a system with 3 domains and 3 mobile wireless stations the number of possible network configurations goes up to 27.

In Table 9.10, $S_2^{M2}(,D^1)$ refers to message stream S_2^{M2} when mobile wireless station S6 is in domain D^1. In the case of S_2^{M3}, both the initiator and the responder are mobile wireless stations, thus S_2^{M3} (D^1, D^3) denotes the case when master M3 belongs to domain D^1 and slave S6 to domain D^3. The cases where the two stations belong to the same domain (IADT) do not need to be considered, since request and response frames are not relayed by the bridges.

In Chapter 7, a formulation which enables the reduction of the pessimism related the number of simultaneous queuing of IDTs requests on a BM was presented. In the case of this example, that formulation is specially suited for IDTs relayed by bridge B3, since the bit rate of the two domains, D^4 and D^3, is somewhat different (0.5 Mbit/s and 2 Mbit/s, respectively). This difference in bit rate permits to guarantee a token cycle time of 0.83 ms and 2.82 ms, for domain D^3 and D^4, respectively. Thus, applying the algorithm presented in Figure 7.6, the number of high-priority messages that can be "simultaneously" queued in M9 will be 2 instead of 4. Table 9.11 takes into account this tighter result.

Table 9.11 – Maximum number of message streams simultaneously queued in a BM

	nh
M4	4
M5	2
M6	4
M7	5
M8	2
M9	2

9.3.2. Computing the WCRT for IDTs

Based on the values provided in Table 9.9 and on the number of high-priority messages which can be simultaneously queued in a BM (Table 9.11), it is now possible to determine the WCRT for IDTs (considering the IDMP inactive). Table 9.12 presents the results, which, in any case, were obtained considering all possible locations of the mobile wireless stations involved in the IDTs.

It is important to note that since M3 is a mobile wireless master, it can belong to domains D^1 and D^3. Consequently, the token cycle times in these domains (which are obtained by $T_{TR}+n\times C_{\sigma}$) is calculated assuming that M3 is in both domains simultaneously (obviously an impossible abstraction). Contrarily, when M3 is involved in a transaction, then the calculation of WCRT for IDTs can be made assuming the real network situation. As an example, when M3 is located in D^1, D^3 only has two active masters, and consequently the token cycle time is reduced from 0.83 ms to 0.56 ms.

Table 9.12 – WCRT for transactions (IDMP inactive)

Stream	$Rbmi_i^k$ (ms)	$Rmlr_i^k$ (ms)
S_1^{M1}	-	3.33
S_2^{M1}	-	3.33
S_3^{M1}	9.69	18.33
S_1^{M2}	12.26	25.09
$S_2^{M2}(,D^1)$	18.07	30.90
$S_2^{M2}(,D^3)$	2.51	15.90
$S_1^{M3}(D^1)$	22.86	32.75
$S_1^{M3}(D^3)$	1.17	8.55
$S_2^{M3}(D^1, D^3)$	14.03	22.75
$S_2^{M3}(D^3, D^1)$	10.14	18.55
$S_3^{M3}(D^1)$	4.24	12.75
$S_3^{M3}(D^3)$	4.24	13.55

Note that in Table 9.12 message streams S_1^{M1} and S_2^{M1} are IADTs and, consequently, their WCRT is much smaller than the WCRT for the other message streams.

9.4. Mobility-related Timings

9.4.1. Phase 1

The duration of Phase 1 mostly depends on the time required to finalise open IDTs by the system BMs. Table 9.13 presents the time span values for the `Start_Mobility_Procedure` message to go from the system GMM to a BM (t_{SMP}^{bm}), and the time required by the `Ready_to_Start_Mobility_Procedure` message to go from a BM to the GMM (t_{RSMP}^{bm}). The time required to finalise the IDTs in each BM of the system (calculated using Eq. (7.7)) is also given individually for every message stream and for every BM.

When calculating the time span t_{RSMP}^{bm} it is possible to reduce the pessimism of this calculation by verifying if the BMs involved on the relaying of the `Ready_to_Start_Mobility_Procedure` message are still required to forward other IDTs. As an example, in the case of M8, since it is the last BM in the system to finalise its IDTs, then the remaining stations in the path to M6 (M9 and M7) do not have any other IDFs to relay. Therefore, it is only necessary to consider that at most one message transaction can be simultaneously queued on BM M9.

Table 9.13 – Phase 1 timings

	t^{bm}_{SMP} (ms)	S_3^{M1} (ms)	S_1^{M2} (ms)	S_2^{M2} (ms)	S_1^{M3} (ms)	S_2^{M3} (ms)	S_3^{M3} (ms)	$t^{bm}_{fin\_IDT}$ (ms)	t^{bm}_{RSMP} (ms)	Total (ms)
M4	5.07	-	-	-	-	-	-	0	4.10	9.17
M5	5.11	-	-	-	30.62	21.78	12.00	30.62	1.19	36.91
M6	0.00	18.01	-	-	-	-	-	18.01	0.00	18.03
M7	0.03	-	-	-	-	17.90	12.80	17.90	0.03	17.96
M8	6.72	-	23.17	28.98	-	-	-	28.98	1.33	**37.03**
M9	6.75	-	-	-	8.92	-	-	8.92	2.40	18.07

The total duration of Phase 1 is equal to 37.03 ms (the higher value is related to BM M8).

9.4.2. Phase 2

The main intervenients on Phase 2 are now the DMMs.

Phase 2 starts when the GMM sends the `Prepare_for_Beacon_Transmission` message, which commands the system DMMs to capture the token. After capturing the token, the DMMs start the inquiry service and send the `Ready_for_Beacon_Transmission` message to the system GMM. Table 9.14, summarises the latencies involved in these operations.

t_{PBT}^{dmm} has been calculated using Eq. (7.6), assuming that M6 (the system GMM) does not have any active message streams, and considering that the BMs on the `Prepare_for_Beacon_Transmission` message IDreq communication path can queue at most one high-priority message: the `Prepare_for_Beacon_Transmission` message (note that IDTs have been disabled during Phase 1).

$t_{cap\_token}^{dmm}$ is equal to the domain cycle time, which can be obtained by Eq. (3.2). Again, in this case it is assumed that M3 is in both wireless domains at the same time. Note that in the case of M8 this time span is equal to 0, since domain D^4 is a wired terminal domain, and therefore it can resume normal operation.

Table 9.14 – Phase 2 timings

	t^{dmm}_{PBT} (ms)	$t^{dmm}_{cap\_token}$ (ms)	t^{dmm}_{RBT} (ms)	Total (ms)
M6	0.00	0.00	0.00	2.24
M5	1.19	0.83	0.23	2.24
M7	0.03	1.09	0.03	1.15
M8	1.28	0.00	0.26	1.54

$t_{RBT}{}^{dmm}$ is calculated using Eq. (7.9). It is assumed that since the network is in inquiry mode, only mobility related messages are transmitted, and therefore BM M4 and M9 can simultaneously queue at most one `Ready_for_Beacon_Transmission` message, related to DMM M5 and M7, respectively. BM M6 must queue two `Ready_for_Beacon_Transmission` messages, one from DMM M8 and another from DMM M7, resulting that the maximum number of message "simultaneously" queued on its output is equal to 2.

The duration of this phase in M6 (the system GMM) is not the sum of the individual delays, but equal to the maximum summation of the Phase 2 timings, since only after receiving the `Ready_for_Beacon_Transmission` message, M6 sends the `Start_Beacon_Transmission` message (which initiates Phase 3).

From the comparison between Table 9.13 and Table 9.14, it is obvious that the main objective of operating the network in inquiry mode is achieved – a substantial reduction on the worst-case response time for IDTs and IADTs conveying IDMP-related messages.

9.4.3. Phase 3

During Phase 3 the system DMMs are commanded by the GMM to start emitting `Beacons`, using the `Start_Beacon_Transmission` message. The `Beacons` are used by the mobile wireless stations to evaluate the quality of the system radio channels.

In this example, only two wireless domains need to be assessed by the mobile wireless stations. The number of `Beacons`, and inherently $t_{beacon}{}^{dmm}$, has been calculated according to the formulations and methods presented in Section 7.3.6. Table 9.15 summarizes the latencies related to the evolution of Phase 3.

Table 9.15 – Phase 3 timings

	t_{SBT}^{dmm} (ms)	t_{beacon}^{dmm} (ms)	t_{phase3}^{dmm} (ms)
M6	0.53	0,00	0.53
M5	0.56	1.25	1.81
M7	0.03	1.18	1.21
M8	0.43	0,00	0.43

$t_{SBT}{}^{dmm}$ has been calculated using Eq. (7.13), assuming no other traffic in the network. The duration of Phase 3 in domain D^2 is only due to the transmission of the `Start_Beacon_Transmission` message. After that, domain D^2 returns to normal intra-domain operation. Also, since domain D^4 is a wired domain (there is no `Beacon` transmission), it returns to normal operation after receiving the `Start_Beacon_Transmission` message.

9.4.4. Phase 4

During Phase 4 the mobile wireless stations are able to enter into a new domain after having assessed the quality of the other radio channels in the system.

The duration of Phase 4 depends on the mobile wireless station being considered, on the wireless domain to where the stations moved, and on the time required before all BMs in the system have updated their routing tables. In the scenario, two mobile wireless stations exist (M3 and S6), which are able to move between domain D^1 and domain D^3. M5 and M7, the DMMs of wireless domains D^1 and D^3, are responsible for detecting the presence of the mobile wireless stations and broadcast updated routing information. Slave stations are detected by `Discovery` messages, and master stations must enter into the destination domain logical ring using the GAP Update mechanism.

Table 9.16 presents the Phase 4 timings, related to discovery sub-phase and to the time required by a master to enter into the logical ring. In this table the notation *station_id*(D^x) represents the case when a station *station_id* is entering into a domain D^x.

Table 9.16 – Phase 4 timings when entering into domains D^1 and D^3

	t_{disc}^{dmm} (ms)	$t_{m\_master\_entry}$ (ms)
M3(D^1)	0.45	5.71
S6(D^1)	0.45	-
M3(D^3)	0.45	7.03
S6(D^3)	0.45	-

The worst-case duration of the discovery sub-phase (t_{disc}^{dmm}) has been calculated using Eq. (7.15). The time required for the entrance of a master station into the logical ring ($t_{m\_master\_entry}$) has been calculated using Eq. (3.4), considering that M3 enters into the logical ring after BM M5 and M9, when located in domain D^1 and domain D^3, respectively.

The last operation of the IDMP is the broadcast of updated routing information, issued by the wireless domains DMMs. Table 9.17 presents those values for the network example. In this table, the notation $t_{RU,station\_id}^{bm}(D^x)$ represents the case where station *station_id* has been detected in domain D^x.

From Table 9.17, it is noticeable that the time required for the `Route_Update` message to reach BM M9, when M3 or S6 are entering into domain D^1, is different. That is due to the fact of assuming, in the case of S6, that M3 belongs to both domains and is already active, therefore the cycle time is higher than in the case of considering that M3 does not belong to the domain.

In the case of the `Route_Update` message related to M3, it is possible to reduce the pessimism by considering that during that time there are no IDTs related to M3 being queued by the BMs in the network.

The same type of reasoning was also made for the transactions involving S6 as a responder, which is translated in a reduction, by one, on the maximum number of IDTs "simultaneously" queued in BMs {M5, M4, M6, M7, M9} when S6 is entering into domains D^3 and D^1. Note that IDTs involving mobile wireless stations are only enabled at the reception of a `Route_Update` message.

Table 9.17 – Phase 4 timings: time required for the `Route_Update` messages to reach the system BMs

	$t_{RU,M3}^{bm}(D^1)$ (ms)	$t_{RU,M3}^{bm}(D^3)$ (ms)	$t_{RU,S6}^{bm}(D^1)$ (ms)	$t_{RU,S6}^{bm}(D^3)$ (ms)
M4	0.03	2.14	0.03	3.12
M5	-	2.17	-	3.14
M6	1.16	0.03	3.12	0.03
M7	1.19	-	3.15	-
M8	3.83	4.30	7.68	5.63
M9	3.80	4.27	7.65	5.60

Table 9.18 presents the total duration of Phase 4 for each BM that is measured from the end of the `Beacon` transmission sub-phase in the wireless domain to which a mobile wireless station has moved to, until a bridge master *bm* changes its routing tables, reflecting the new configuration of the network. In that table the notation $t_{phase4}^{station\_id,bm}(D^x)$ represents the total duration of Phase 4 for a BM *bm* related to a station *station_id* that is entering into domain D^x.

Table 9.18 – Phase 4 total time duration

	$t_{phase4}^{M3,bm}(D^1)$ (ms)	$t_{phase4}^{M3,bm}(D^3)$ (ms)	$t_{phase4}^{S6,bm}(D^1)$ (ms)	$t_{phase4}^{S6,bm}(D^3)$ (ms)
M4	6.20	9.90	0.48	3.57
M5	6.20	9.93	0.45	3.60
M6	7.33	7.79	3.57	0.48
M7	7.36	7.76	3.60	0.45
M8	10.00	12.06	8.13	6.08
M9	9.97	12.03	8.10	6.05

Table 9.18 clearly shows that the time required for a master to enter into the logical ring is higher than for a slave. That is due to the GAP Update mechanism used by the master to enter into the logical ring. This value is even higher if the domain is setup with a higher `Gap Update` factor. For example, if a `Gap Update` factor of 100 (as recommended by the PROFIBUS-DP standard) was used, then the time required before BM M9 updates its routing tables, when M3 enters into domain D^1 ($t_{phase4}^{M3,M9}$), would be equal to 39.67 ms.

9.5. IDMP Impact on the WCRT of Transactions

After having calculated the duration of the four phases of the IDMP it is possible to determine the impact of the IDMP on the WCRT of IDTs and IADTs. The first part of this section presents the calculation of the inaccessibility periods, during which the various types of transactions are disabled, and also the periods related to the inaccessibility of mobile wireless stations. These are then considered in the WCRT analysis of the system message streams.

9.5.1. Inaccessibility Periods

The time during which IADTs are disabled in a domain ($t_{IADT\_dis}{}^{dmm}$) can be calculated using Eq. (8.1) or Eq. (8.2). Table 9.19 presents the obtained results.

It is interesting to note that since the GMM has a "central" location in the network, the variability of this time span between domains is small. Also note that, in this case, the differences between this time span in wired and wireless domains is also small, due to the reduced duration of Phase 3.

Since domain D^4 is a wired terminating domain, it returns to normal intra-domain operation after the reception of the `Prepare_for_Beacon_Transmission` message. Therefore, there is no inaccessibility period for D^4. This is an interesting characteristic, especially in the case of a mobile wired domain (e.g. an AGV), since it permits to maintain intra-domain tight control loops.

Table 9.19 – Time during which IADT are disabled in a domain

Domain of	$t^{dmm}_{IADT\_dis}$ (ms)
M6	2.49
M5	2.77
M7	2.77
M8	0.00

The time during which IDTs are disabled in a BM represent the main impact of the IDMP protocol on the delays associated with IDTs. Table 9.20 presents those relevant values, which were calculated using Eq. (8.3) and Eq. (8.4).

Table 9.20 – Time during which IDTs are disabled in a BM

BM	$t^{bm}_{IDT\_dis}$ (ms)
M4	34.73
M5	36.44
M6	34.73
M7	39.70
M8	33.44
M9	34.19

It is interesting to note that this time span is larger for BMs connected to wireless domains and it is smaller for the wired domains more distant from M6 (the system GMM).

Each BM inhibits transactions with mobile wireless stations, from the reception of the `Start_Mobility_Procedure` message until receiving `Route_Update` messages, regarding those stations. Table 9.21 presents the time span during which IDTs addressed to mobile wireless stations are disabled in a BM, which can be calculated using Eq. (8.5) and Eq. (8.6), for a mobile wireless master and for a mobile wireless slave, respectively.

Table 9.21 – Time during which IDT addressed to mobile wireless stations are disabled in a BM

	$t_{IDT\_MM\_dis}^{bm,M3}$ (ms)	$t_{IDT\_MS\_dis}^{bm,S6}$ (ms)
M4	45.31	38.99
M5	45.31	38.99
M6	48.85	44.66
M7	48.85	44.66
M8	45.83	42.50
M9	45.77	42.44

In Table 9.21, as expected, the time during which IDTs addressed to mobile wireless slaves are disabled in a BM is smaller than the case of mobile wireless masters, which is due to the use of the Gap Update mechanism by the mobile wireless masters.

Mobile wireless stations are not capable of transmitting or receiving messages from the time at which the domain DMM captures the token until the end of the discovery sub-phase or its entrance into a new logical ring, for slaves and masters, respectively. Table 9.22 gives the worst-case scenarios for this particular inactivity period, which have been calculated using Eq. (8.7). As expected, the values are larger for mobile wireless master stations.

Table 9.22 – Time during which mobile wireless stations are inaccessible

	Value (ms)
$t_{ina\_D^1 \to D^3}^{M3}$	13.50
$t_{ina\_D^3 \to D^1}^{M3}$	12.93
$t_{ina\_D^1 \to D^3}^{S6}$	7.52
$t_{ina\_D^3 \to D^1}^{S6}$	11.06

9.5.2. WCRT for the System Message Streams

Based on the results presented in the previous sections, it is now possible to compute the WCRT for the example message stream set considering the effect of the IDMP. Table 9.23, summarises those results.

Streams S_1^{M1} and S_2^{M1} are related to IADTs. Their WCRT was obtained using Eq. (8.8).

Streams S_3^{M1} and S_1^{M2} are related to IDTs involving resident wireless stations or wired stations. The first step in calculating the WCRT for these IDTs depends on whether there is the possibility for the initiator to queue requests during the evolution of the IDMP phases or during its periods of inaccessibility. For details consult Section 8.3.2.

In the cases of S_3^{M1} and S_1^{M2}, since $t_{IADT\_dis}^{bm} < T_1^k < t_{IDT\_dis}^{bm}$, then the WCRT can be calculated using Eq. (8.11).

The calculation of these two message streams $Rbmi_i^k$ values must be made using Eq. (8.12), but only S_1^{M2} is affected by additional delays.

Table 9.23 – WCRT for the system message streams

Stream	$Rbmi_i^k$ (ms)	$Rmlr_i^k$ (ms)	$Rmlr\_m_i^k$ (ms)
S_1^{M1}	-	3.33	6.10
S_2^{M1}	-	3.33	6.10
S_3^{M1}	9.70	18.33	61.39
S_1^{M2}	12.26	25.90	76.94
$S_2^{M2}(,D^1)$	18.07	30.90	98.49
$S_2^{M2}(,D^3)$	2.51	15.90	73.49
$S_1^{M3}(D^1)$	22.86	32.75	88.87
$S_1^{M3}(D^3)$	1.17	8.55	59.00
$S_2^{M3}(D^1, D^3)$	14.03	22.75	78.56
$S_2^{M3}(D^3, D^1)$	10.14	18.55	74.93
$S_3^{M3}(D^1)$	4.24	12.75	63.08
$S_3^{M3}(D^3)$	4.24	13.55	59.00

The other streams involve mobile wireless stations. Message stream S_2^{M2} involves a wired master and a mobile wireless slave and its WCRT can be calculated by Eq. (8.13). Streams S_1^{M3} and S_3^{M3} involve a mobile wireless master and a wired slave. Consequently its WCRT can be calculated by Eq. (8.14).

Finally, message stream S_2^{M3} involves two wireless stations. Therefore, its WCRT can be calculated by using Eq. (8.16).

As it can be observed, there is a significant impact of the IDMP on the WCRT of the message streams. In some cases the resulting value is almost the 3 times the value without considering the IDMP.

9.6. Results Extracted from Simulation

The results obtained in the last sections are based on worst-case analytical formulations provided in Chapters 3, 7 and 8. As all worst-case analysis, it usually leads to pessimistic results. Nevertheless, these results can be trustworthy for the setting of the timing parameters in the proposed hybrid wired/wireless PROFIBUS network. Another approach for characterising the timing behaviour of the network is the use of statistical data provided by simulation or by a stochastic timing analysis of the network. This kind of results can be used for the setting of the network when running soft real-time applications, usually leading to a higher utilization of the available bandwidth. Additionally, the development of the simulation tool enables the validation of the proposed protocols on a controller environment.

In this section we present some results which were obtained by the simulation of the network architecture proposed in this thesis using the standard simulation package OMNet++ (Varga, 2005). This package is a modular discrete event simulator, where the system components can be built using C++ and assembled into larger components and models using a high level language called NED.

9.6.1. Basics on the Simulator Architecture

The simulator platform (Sousa and Ferreira, 2005) is composed by six main modules: `Controller`, `HW2PSim`, `Domain`, `Bridge`, `Master` and `Slave`.

The `HW2PSim` module is a compound module that represents the network and contains the instances of the remaining modules.

The `Controller` module main responsibilities are related to the configuration of the other network modules and the coordination of the data output to files recorded by the `Master` and `Slave` modules. This data includes the message stream response times as also data related to the operation of IDMP agents, e.g. the GMM, DMMs and BMs state transitions. Additionally, this module is also responsible for the network configuration changes, which support the mobility of the mobile wireless stations.

The OMNet++ simulator only implements one-to-one connections, therefore a `Domain` module has been developed in order to allow the broadcast of messages between stations belonging to the same domain. This module is responsible for the simulation of the channel, transmission delay and the broadcast delivery of messages to all modules.

The `Master` and `Slave` modules are composed by three other sub-modules: `Master_PHY` (`Slave_PHY`), `Master_DLL` (`Slave_DLL`) and `Msg_Stream`. The module `Master_PHY` implements the master PhL functionalities. In this particular case, this module only forwards the frames relayed by the DLL, but it can be used to simulate the occurrence of errors or the coding of the frames. The `Master_DLL` module implements the main functionalities related to the PROFIBUS DLL. Additionally, it also implements the functionalities required by the IDP and IDMP on the BMs, and the IDMP-related functionalities provided by the DMMs and the GMM. The `Msg_Stream` module can be configured for periodically requesting services to the DLL. This module can, in the future, be replaced by a fully functional Application Layer module, like PROFIBUS-DP.

Figure 9.2 depicts, for the example network of Figure 9.1, its representation using the modules mentioned above. The `Controller` module is able to communicate with all other modules for configuration and controlling purposes. The `Master` and `Slave` modules are connected to a `Domain` module, symbolized by a rectangle for the case of wired domains, or by a cloud for the case of wireless domains. The `Bridge` modules connect the two BMs of a bridge.

The simulator is also capable of emulating some of the network parameters which are not constant, like the station delay of responders (T_{SDR}) and the message stream timing parameters (period, initial offset and frame size). These parameters can be made variable according to a probabilistic function with parameters chosen by the user.

An additional tool, the Timeline Visualisation Tool, has also been developed, which allows the visual representation of the frames transmitted in the network, and respective time latencies. This tool is of paramount importance on the protocol verification, and helps on detecting some particular cases related to the response time histograms, as detailed later in this chapter.

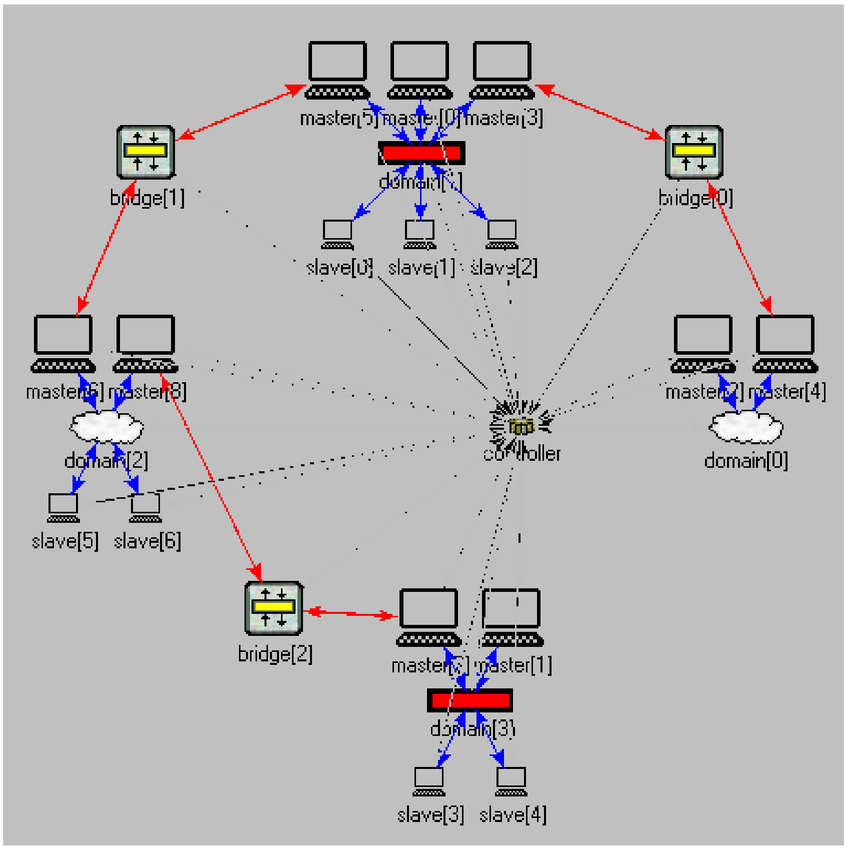


Figure 9.2 – Representation of the network configuration by the simulator

9.6.2. Simulation Parameters and Operational Characteristics Assumptions

A PROFIBUS standard master is usually a dedicated device composed by a communication module (mostly in hardware) and a CPU module running the control software. The master stations used in our simulation have been modelled according to the following operational characteristics assumptions:

- the variability of the master timings parameters is usually reduced, as confirmed by some experimental measurements (Behaeghel *et al.*, 2003);
- it is expected that the clocks of the master stations in the system may have some drift between them;
- the masters are not synchronised between them;
- each master is allowed one retry, as defined by the PROFIBUS-DP default parameters.

These assumptions were applied to the simulator by setting the offset and the period of the message streams using probabilistic variables. Therefore, the message stream offset has been set using a triangular distribution function with minimum equal to 0 ms, maximum equal to 5.0 ms and an apex at 2.5 ms. The message stream period has also been set using a triangular distribution function with minimum equal to 4.8 ms, maximum equal to 5.2 ms and an apex at 5.0 ms.

For the timing behaviour of the slave stations it has been assumed that the responder delay (T_{SDR}) is variable. Therefore, it has also been set using a triangular distribution function with an apex at 70 bit time and extremes at 11 and 100 bit times for all the domains. This triangular distribution function has been chosen since according to the PROFIBUS-DP specification the T_{SDR} parameter value can vary between 11 and 100 bit times on a network operating at 1.5 Mbit/s and at 0.5Mbit/s.

The other time parameters of the simulation have been setup with the parameters presented in Section 9.2, assuming that the mobility procedure period is equal to 200 ms.

The simulation results have been obtained as the aggregate result of 10 runs, each with 60 s of duration, and using a different seed value in order to improve the randomness of the data. Additionally, in order to be able to compare the results of the simulation with the results obtained by the analytical formulations presented before, the simulations had been made independently for each set of master message streams, as follows:

- for the master to whom we want to perform the measurements, the message stream periods were set to a constant value of 5.0 ms;
- for the other masters, the message streams parameters were set according to the period and offset values proposed above.

This kind of configuration allows the creation of a higher number of different traffic conditions in the network.

9.6.3. Simulation Results

This section presents and analyses some representative response time histograms for the message streams defined in Table 9.8.

Figure 9.3 depicts, using a logarithmic scale, the number of measured response times in a specific range, related to message streams S_1^{M1} and S_2^{M1}, which convey IADTs. In the specific case of these message streams, the minimum measured response time is equal to 0.27 ms and 0.64 ms, for S_1^{M1} and S_2^{M1}, respectively. The maximum response time is equal to 2.60 ms and 3.27 ms, for S_1^{M1} and S_2^{M1}, respectively. Using the analytical approach proposed in Chapter 8, the WCRT obtained for message streams S_1^{M1} and S_2^{M1}, is equal to 6.10 ms.

In Figure 9.3 it is interesting to note that message stream S_2^{M1} has a higher probability of having a response time higher than for message stream S_1^{M1} (i.e. the histogram related of S_2^{M1} is more shifted to the right than the histogram of S_1^{M1}). The main reason is due to the implementation of the simulator. In our implementation S_1^{M1} is always queued first than S_2^{M1}, and both have the same period. Therefore, frames related to message stream S_2^{M1} always have to wait for the transmission of frames related to message stream S_1^{M1} before begin transmitted.

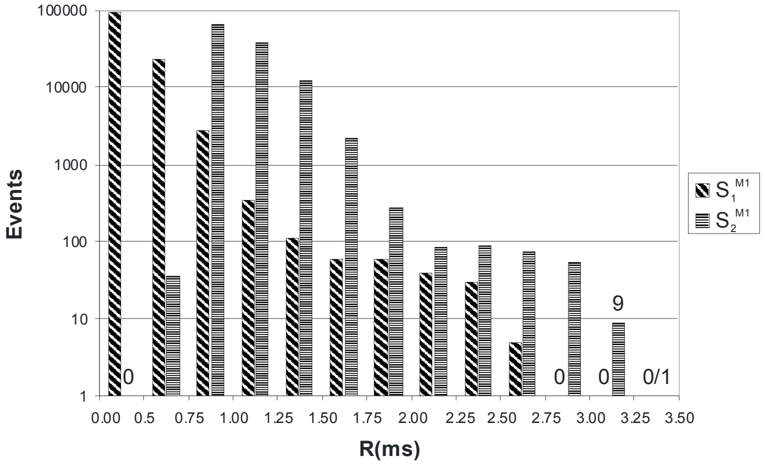


Figure 9.3 - Response time histogram for message streams S_1^{M1} and S_2^{M1} (with IDMP)

Figure 9.4 presents, for the same message streams depicted in Figure 9.3, the results of the simulator with the IDMP disabled.

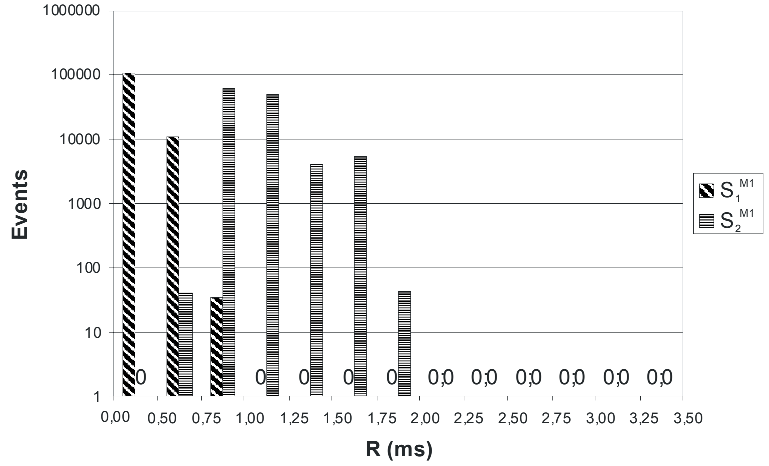


Figure 9.4 - Response time histogram for message streams S_1^{M1} and S_2^{M1} (without IDMP)

Comparing the histogram depicted in Figure 9.3 with the same scenario without mobility (Figure 9.4) it is noticeable that in this case there is a smaller variability of the response time values, and that the maximum response time is smaller (0.89 ms and 1.95 ms for S_1^{M1} and S_2^{M1}, respectively). The minimum response times are identical. In fact, the IDMP leads to higher response times and increases the variability of the measured response times. Therefore, in the case of Figure 9.3, the results higher than 2.00 ms are most probably due to the effects of the IDMP. Nevertheless, these results only represent 0.26% of the total. To confirm these situations we had resorted to the timeline created by the Timeline Visualisation Tool, which confirmed our assumptions.

Figure 9.5 depicts a histogram related to the response for message stream S_2^{M2}, in which wired master M2 is the initiator and the mobile wireless slave S6 is the responder.

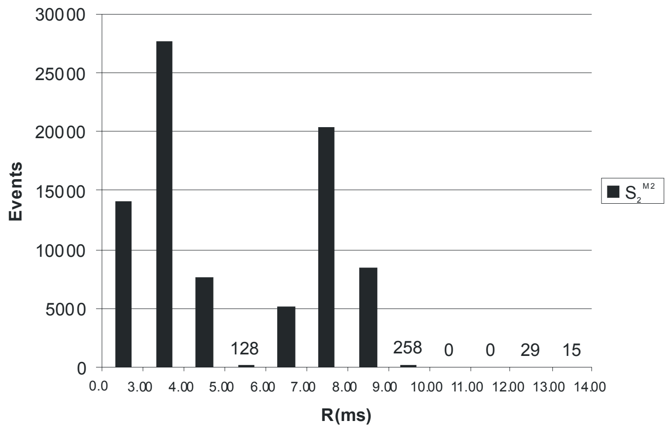


Figure 9.5 - Response time histogram for message stream S_2^{M2}

At a first look this histogram can be a bit surprising, since about 60% of the measured response times are below 5 ms. Note that the message stream period (T_2^{M2}) is equal to 5 ms, and also that the IDP operation requires that, in the case of an IDT, the initiator, after making the first request, should subsequently repeat the same request before obtaining a response. Intuitively, the response time for S_2^{M2} should be higher than 5 ms. In our measurements, 65.2% of the response time values are related to the situation when S6 is in domain D^3 and 34.8% to the case when S6 is in D^1.

What happens in this particular case is that, since the simulator was configured to make one DLL retry, then M2 makes one DLL retry after transmitting the first request. The first request is accepted by the IDT BM_{ini} (in this case BM M8), which obtains a response from S6 before the completion of the DLL retry (this situation happens only when S6 is on domain D^3). This is a consequence of the small amount of traffic in the network, and of the fact that the bit rate in domain D^4 (to which M2 belongs) is considerably slower than the bit rate in domain D^3.

In Figure 9.5, the cases when the response time is larger than 10.00 ms occur by the combination of two factors: S6 is in domain D^1 and the IDT is affected by the operation of the IDMP. The response time values between 5 ms and 10 ms are mostly related to the cases when S6 is in domain D^1 and in a minor percentage when S6 is in domain D^3. These cases have also been confirmed by the use of the Timeline Visualisation Tool.

The minimum response time for message stream S_2^{M2} is equal to 2.17 ms and the maximum to 13.68 ms. Comparing these results with the results for the WCRT (88.49 ms), it is easy to conclude that there is probably more room for improvement on the formulation of the WCRT for IDTs involving mobile stations (in terms of reducing the pessimism). Note, however, that the simulation does not necessarily provide the real worst-case.

Finally, Figure 9.6 depicts a histogram related to the response time of message stream S_2^{M3}, an IDT between a mobile wireless master M3 and a mobile wireless slave S6, the two mobile stations in the system. Note that this transaction is an IADT when M3 and S6 are on the same domain.

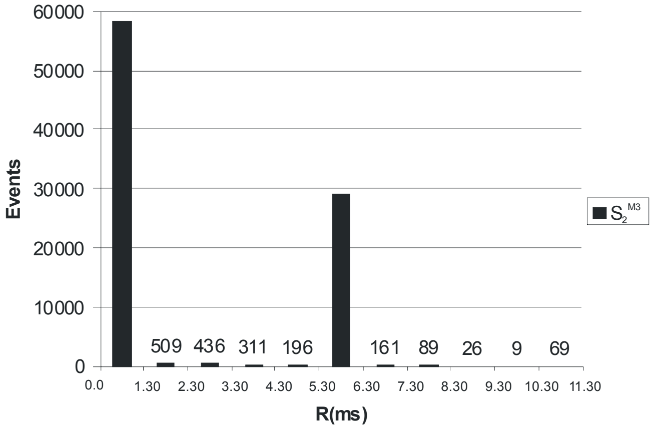


Figure 9.6 - Response time histogram for message stream S_2^{M3}

In our simulation model, the inter-domain mobility of the mobile wireless stations assumes the following conditions: M3 changes from one wireless domain to the other every 200 ms, and S6 changes from one wireless domain to the other every 400 ms. Consequently, the two stations alternate between being in the same wireless domain and on different wireless domains.

When M3 and S6 are on the same wireless domain, then the response time varies between 0.38 ms (also the minimum response time) and 1.30 ms. In Figure 9.6 it is also noticeable another histogram peak located on the interval between 5.30 ms and 6.30 ms, which occurs when the two stations are located on different domains, as also the events related to the higher response times. The maximum value registered is equal to 11.21 ms, which happens when the stations are on different domains and M3 must make two request repetitions before receiving a response related to message stream S_2^{M3}. In our analytical worst-case model, the corresponding value is equal to 78.56 ms.

9.7. Comparing with the Repeater-based Approach

In Chapter 1, we have defended that the main advantages of the bridge-based approach over the repeater-based approach relies on the fault isolation between domains, on the more favourable setting (independently for each domain) of the `Slot Time` parameter and on the fact that no inserted idle time is necessary. These advantages enable a better responsiveness to errors and smaller latencies for IADTs.

This section compares the worst-case timing analysis results for both types of networks when some network parameters are changed, highlighting the advantages and the disadvantages of the bridge-based approach.

9.7.1. Equivalent Repeater-based Network Topology

A repeater-based network that is equivalent to the example network being used in this chapter (Figure 9.1) is presented in Figure 9.7. Instead of the bridges, this network comprises two Structuring & Linking Intermediate Systems (SLIS) – {SLIS1 and SLIS2}, which also include the base station functionality, therefore structuring wireless domains D^1 and D^3. Wired domain D^4 is connected to wireless domain D^3 by a Linking Intermediate System (LIS) – {LIS1}. The network comprises the same stations (except for the absence of the BMs), but additionally it includes the Mobility Master (MM) station, which is responsible for periodically triggering the mobility management procedure. The set of message streams is the same as the one presented in Table 9.8.

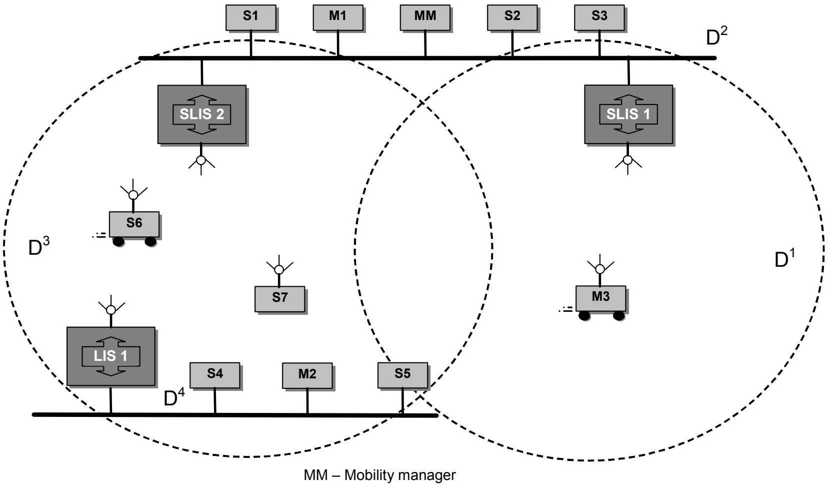


Figure 9.7 – Repeater-based version of the example network

The network parameters are similar to the parameters defined for the bridge-based network in Tables 9.2, 9.3, 9.5 and 9.6.

The main differences on the parameters are related to `Idle Time` and `Slot Time` parameters, since both depend on the maximum size of the frames relayed by the repeaters, the number of repeaters in cascade and the bit rate in each medium. The `Slot Time` parameter must be set according to Eq. (2.6), and the `Idle Time` parameters by the methodology briefly overviewed in Chapter 2 and detailed in (Alves, 2003). It was also considered that the internal forwarding delay of the repeaters is equal to 30 μs, except if otherwise noted. The parameters related to the `Beacon` transmission are equivalent to the bridge-based version of the network, but defined by the proper formulation for the repeater-based approach.

The referred parameter values were calculated with the help of a computing tool, the RFieldbus System Planning software, which is described in (Behaeghel *et al.*, 2003). These settings insure that both networks are equivalent and that the timing analysis results can be compared.

WCRT Results Comparison

Table 9.24 contains the results for the message streams' WCRT for both the bridge-based and the repeater-based approach. The WCRT for the repeater-based approach was obtained by means of the formulation proposed in Eq. (3.23), assuming that the duration of the mobility period (t_{mob}) is equal to 1.87 ms, and that the duration of the `Beacon Trigger` message (t_{BT}) is equal to 0.12 ms in a wired domain. The value for t_{mob} and for t_{BT} was calculated with the support of the RFieldbus System Planning software. The values presented for the bridge-based approach are those obtained previously in this chapter.

Table 9.24 – WCRT for the system message streams

x	***Bridge***	***Repeater***
Stream	$\boldsymbol{Rmlr\_m_i^k}$ **(ms)**	$\boldsymbol{Rslr_i^k}$ **(ms)**
S_1^{M1}	6.10	28.45
S_2^{M1}	6.10	28.45
S_3^{M1}	61.39	29.29
S_1^{M2}	76.94	19.49
S_2^{M2}	88.49	19.74
S_1^{M3}	88.87	29.43
S_2^{M3}	78.56	28.84
S_3^{M3}	63.08	28.59

The most obvious conclusions which can be withdrawn from the results in the table are that, in this particular scenario, the bridge-based approach provides smaller WCRT for message streams associated with IADTs (S_1^{M1} and S_1^{M2}) and a larger WCRT for message streams associated with IDTs, as expected.

9.7.2. Variability of the WCRT

In both approaches the WCRT varies as a function of the network domains bit rate, maximum frame size and ISs delays. A reduction on the domain bit rate is translated into larger message cycle latencies. Larger maximum frame sizes also increase the WCRT of a message transaction, since there is an increase on the domain cycle time, which is calculated by Eq. (3.2). Additionally, the latencies on the ISs obviously increase the WCRT of IDT. The rest of this section highlights the differences, in terms of timing behaviour, between the two kinds of approaches when the above referred parameters are varied.

Variability of the WCRT as a Function of the Bit Rate

One advantage of the bridge-based approach is that the different network domain parameters can be independently set (e.g. the bit rate) and that this setting generally has a small influence on the overall system performance. Figure 9.8 compares the WCRT of message stream S_1^{M1} and message stream S_1^{M2}, in both approaches, assuming the network parameters defined in Section 9.7.1 and varying the bit rate in domain D^4. In the figure, and in the remaining figures in this chapter, a B before the message stream specifies that the values are related to the bridge-based approach whereas a R before the message stream specifies that the values are related to the repeater-based approach.

In this comparison, we are considering that the bit rate in domain D^4 varies between 187.5 Kbit/s and 5 Mbit/s. Consequently, the network parameters T_{SL}, T_{ID1}, T_{ID2} and t_{mob} must be recalculated for every bit rate. In the case of the repeater-based approach these changes affect all stations in the network. In the case of the bridge-based approach, these changes only affect stations belonging to domain D^4.

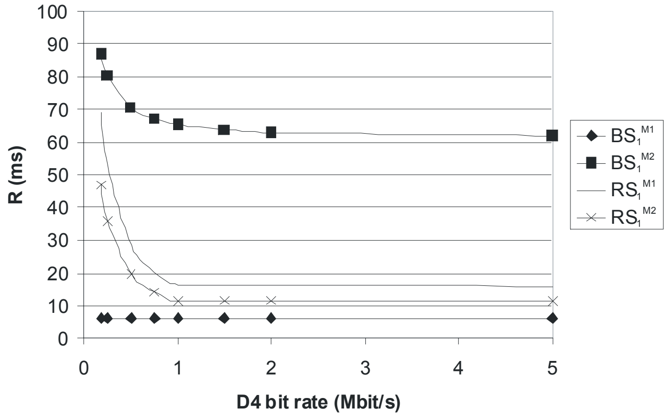


Figure 9.8 – Influence of D^4 bit rate on the WCRT for IADTs and IDTs

Message stream S_1^{M1} is an IADT between master M1 and slave S1, both belonging to domain D^2. S_1^{M2} is an IDT between M2 and S3, which belong to D^4 and D^2, respectively.

With the bridge-based approach, varying the bit-rate on domain D^4 does not influence the WCRT of message stream S_1^{M1} (calculated by Eq. (8.8)), since a change on domain D^4 bit rate only influences the time required by M8 to capture the token ($t_{cap\_token}^{dmm}$), which is subtracted from the duration of Phase 2 on the equation which defines the time during which IADT are disabled in a domain, Eq. (8.1) and Eq. (8.2). As a consequence, in this particular situation the WCRT of message stream S_1^{M1} remains unchanged despite varying the bit rate in domain D^4.

In the case of message stream S_1^{M2}, the main influence is due to the increase on the WCRT of message transactions which are relayed through domain D^4 and on the increase of the inaccessibility periods related to the IDMP, specifically on the time during which IDTs are disabled in a wired or wireless BM, $t_{IDT\_dis\_wr}^{BM}$ and $t_{IDT\_dis\_wr}^{BM}$, respectively.

In the case of the repeater-based network, it is noticeable that the decrease on domain D^4 bit rate strongly influences the WCRT for both message streams. The main reason for this behaviour is related to the fact that a repeater-based network operates in broadcast mode, therefore it is necessary to increase the inserted idle time on the ISs to compensate for the latencies related to the transactions involving stations in domain D^4.

WCRT Variability as a Function of the Maximum Frame Size

The variation on the size of the request and response frames related to a message stream not only influences the calculation of its WCRT, but also the WCRT of the other message streams in the network (see Section 3.2 for details). As an example, the calculation of the token cycle time (T_{cycle}^k) parameter depends on the maximum latencies of a message cycle.

In the case of the bridge-based approach, an increase on the maximum size of a message stream increases the token cycle time associated with the domains which relay the frames related to that message stream. In the case of the repeater-based approach that increase not only augments the token cycle time in the overall network, but also forces the increase on the `Slot Time` and `Idle Time` parameters. These increased latencies imply a larger WCRT for the system message streams.

Figure 9.9 compares the results of the two approaches for message stream S_1^{M1} and S_1^{M2}.

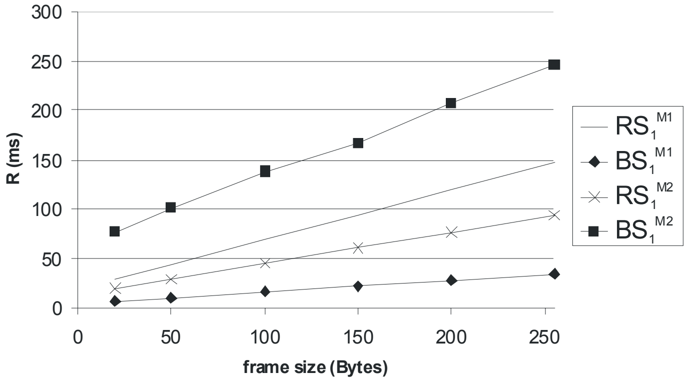


Figure 9.9 – Influence of the maximum frame size on the WCRT

In the case of Figure 9.9, it has been considered that there is another message stream which increases their maximum size of the frames (e.g. S_2^{M1}). Message stream S_1^{M1} is an IADT between wired master M1 and wired slave S1. S_1^{M2} is an IDT between M2 and S3. In the case of the bridge-based approach it has been considered that the increase on the maximum message size only affects domain D^2.

From Figure 9.9, it is obvious that the increase on the maximum frame size has similar effects in case of message streams S_1^{M2} in the repeater-based and in the bridge-based approach.

In the repeater-based approach the increase on the S_1^{M1} WCRT is mainly due to the increase on the `Idle Time` parameters required to encompass the increase on the maximum frame size. In the bridge-based approach the increase on the S_1^{M1} WCRT is due to the increase on the token cycle time in D^2, and also on the time during which IADTs are disabled in D^2. Therefore it can be concluded that on the bridge-based approach the effect of increasing the maximum frame size in a domain translates into larger response times for transactions that involve stations belonging to that domain and has minor effects on the other message transactions (only due to the increased latencies of the IDMP). In the repeater-based approach, increasing the maximum frame size affects the WCRT for all message transactions.

WCRT Variability as a Function of the ISs delays

Another aspect which influences the timing behaviour of the network is related to the delays imposed by the ISs, either repeaters or bridges. Figure 9.10, presents the WCRT for message transaction S_1^{M1} and S_1^{M2}, as a function of the ISs delays.

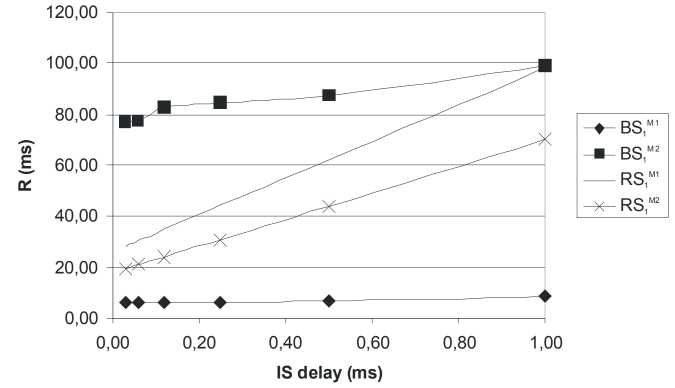


Figure 9.10 – Influence of the IS delay on the WCRT

In the case of the bridge-based approach, the ISs (bridge) delay has a small effect on the WCRT of message stream S_1^{M1} (an IADT), since the frames exchanged in this kind of transactions are not relayed by bridges. The small increase is due to the increase of the IDMP-related latencies, specifically on the time during which IADTs are disabled in domain D^2.

Also in the case of message stream S_1^{M1}, the effects are small and are mainly due to the increase on the IDMP latencies. Also note that the increase on the bridge delay is responsible for an increase on $Rbmi_i^k$, which does not translates into additional latencies since the WCRT given by Eq. (7.1) depends mostly on the number of attempts (A_i^k) made by the transaction initiator until receiving a response from BM_{ini}, which is calculated by Eq. (7.2). Due to use of the ceiling function in that equation, a small variation in $Rbmi_i^k$ does not alter the number of attempts required for retrieving a response from BM_{ini} by the transaction initiator, and therefore there is no increase on the WCRT for the message stream.

In the case of the repeater-based approach, the ISs delay affects significantly both message streams, since it is required to increase the `Idle Time` and `Slot Time` parameters.

9.7.3. Comparing the Responsiveness to Errors in Both Approaches

As described before, one of the major problems with the repeater-based approach is that the setting of the `Slot Time` parameter must be made to much larger values than on the bridge-based approach. Larger values for the `Slot Time` parameter imply a lower responsiveness of the network to token errors and to transmission errors, since the time required to detect an error is increased. Consequently, the time required by the PROFIBUS DLL before making another retry also increases. Additionally, it is expected that the occurrence of errors becomes higher in a wireless domain than in a wired domain, which makes this problem more acute for the type of network being considered.

Another consequence of setting a high `Slot Time` parameter is related to the time required to recover from a lost token situation. In PROFIBUS, a lost token is detected when a master does not detect any network activity for a time defined by the setting of its `Time Out` timer (T_{TO}), which is set as T_{TO}=6*T_{SL}+2*n*T_{SL}, where n is the master address. In the network example being discussed, the `Time Out` setting would be equal to 1.15 ms and 29.82 ms, in the bridge-based version and in the repeater-based version, respectively.

Another aspect which influences the responsiveness to errors of the network is how the WCRT of a message transaction increases due to the need of performing several DLL retries before obtaining a response. In the case of IADTs, this timing is influenced mostly by the `Slot Time` parameter value.

The rest of this section analyses the variability of the `Slot Time` parameter value as a function of the bit rate, maximum frame size and IS delay.

Slot Time Variability as a Function of the Bit Rate

Varying the bit rate in domain D^4 requires the recalculation of the `Slot Time` parameter. In the case of the bridge-based approach, it only affects domain D^4. Since this parameter is defined in bit times, a change in the bit rate is results into a change in the `Slot Time` value. Contrarily, in the repeater based-approach, the `Slot Time` parameter depends not only on the bit rate but also on the system turnaround time. Additionally, its setting is common to all stations in the network.

Figure 9.11 depicts the results for the `Slot Time` parameter as a function of the bit rate in domain D^4. It is noticeable that in the repeater-based approach the `Slot Time` is much larger than in the bridge-based approach. This directly implies additional latencies in the WCRT of all message streams in the system. This aspect is particular evident when the bit rate of D^4 is rather small. Also note that an increase in the `Slot Time` parameter results in a lower responsiveness to errors.

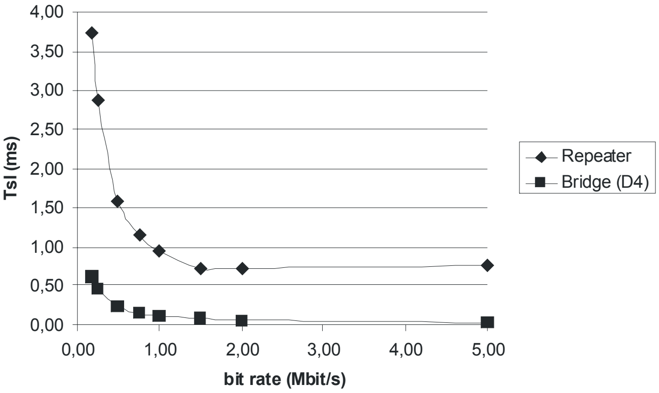


Figure 9.11 – Influence of D^4 bit rate on the `Slot Time`

Slot Time Variability as a Function of the Maximum Frame Size

Figure 9.12 depicts the influence of the maximum frame size on the `Slot Time` parameter value.

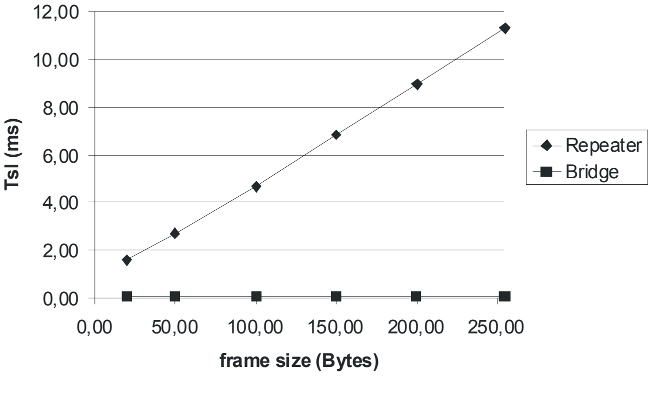


Figure 9.12 – Influence of the maximum frame size on the `Slot Time`

In the repeater-based approach, the maximum size of a frame also has a strong influence on the `Slot Time` parameter, due to the increase on the worst-case queuing delays on the repeaters. Contrarily, in the case of the bridge-based solution, the `Slot Time` can be set to a much smaller value and it does not depend on the maximum frame size as it can be observed in Figure 9.12.

Influence of the number of retries on the WCRT

Figure 9.13 depicts the impact on the WCRT of message streams S_1^{M1}, S_3^{M1}, considering the possibility of executing several retries during the execution of a message transaction associated with those streams. In this scenario, it is assumed that errors, leading to the execution of retries, only occur on transmissions performed in domain D^3.

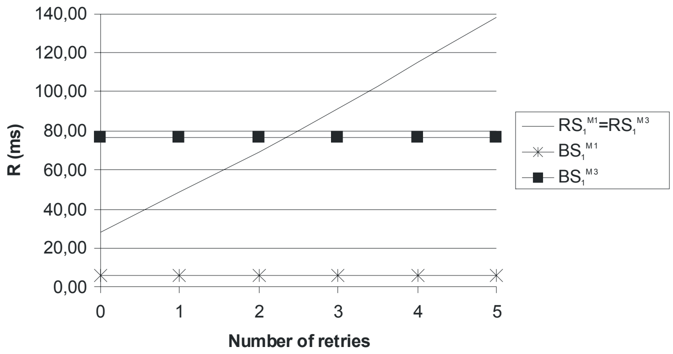


Figure 9.13 – Influence of the number of retries on the WCRT

In the case of the bridge-based approach, it is noticeable that the loss of frames in domain D^3 does not have any influence on the WCRT of message streams S_1^{M1} and S_3^{M1}.

Message stream S_1^{M1} is an IADT between stations in domain D^2. Therefore, due to the isolation provided by the bridge-based approach, the events in domain D^3 will not influence the IADTs associated with S_1^{M1}.

The case of message stream S_3^{M1} (with station M1 as its initiator and station S5 as its responder) requires a more careful reasoning. Its transactions require the transmission of IDFs through domain D^3. Therefore, the delay experienced by the IDT, from the reception of the request at the BM_{ini} until the arrival of the respective response ($Rbmi_i^k$), is influenced by the number of retries performed in domain D^3. Nevertheless, the WCRT given by Eq. (7.1) depends mostly on the number of attempts made by the transaction initiator until receiving a response from BM_{ini}, which is calculated by Eq. (7.2). Due to use of the ceiling function in that equation, a small variation in $Rbmi_i^k$ does not alter the number of attempts required for getting a response from BM_{ini}, and that is exactly what happens in the present scenario.

In the case of the repeater-based approach, the response time increases linearly with the number of retries. When the number of retries is set to 3 or higher, then the WCRT on repeater-based approach becomes larger than on the bridge-based approach.

9.8. Summary

This chapter presented a numerical example which illustrates in detail how the timing analysis proposed in Chapters 3, 7 and 8 can be instantiated to an actual network system. Additionally, this chapter also compares the results obtained by the analytical formulations of the WCRT with results obtained by simulation, therefore validating the proposed protocols and timing analysis. To finalise the chapter, the proposed bridge-based approach is compared with the repeater-based approach, presenting the advantages and disadvantages of both under diverse operating scenarios.

Chapter 10

Conclusions and Future Work

This chapter reviews the research context and objectives of this thesis, resumes the most relevant contributions and highlights the possible directions of future work in the field of wireless networks for industrial automation.

10.1. Research Context and Objectives

Nowadays, industrial automation applications collect large benefits from the use of fieldbuses for the interconnection of distributed devices. Usually, these systems are supported by a wired infrastructure. There is now an increased pressure to extend the capabilities of fieldbuses with wireless communication functionalities, in order to support mobile devices. Wireless communication systems are of particular interest on supporting mobile machine parts, mobile vehicles and temporary or frequently reconfigured production lines, for example.

The users of such networks also expect the level of performance of the wireless extensions to be at least similar to those existing in wired fieldbus protocols. Thus, it is of paramount importance that these networks provide the same levels of throughput, reliability and real-time performance.

The use of wireless communications in industrial automation applications also creates new challenges. As an example, it is necessary to provide ease of interoperation between wireless devices and legacy (wired) devices. Also, wireless devices have a limited radio coverage area, which demands that the factory-floor is divided into several wireless cells. Therefore, devices like Automatic Guided Vehicles (AGV) or portable monitoring apparatus require specific mechanisms enabling them to move seamlessly between different wireless cells.

The work in this thesis was triggered by the RFieldbus project, where a hybrid wired/wireless PROFIBUS-based network architecture has been developed. In RFieldbus, the interconnection between wired and wireless domains is supported by Intermediate Systems (IS) operating as repeaters, since such solution does not involve changes on the higher PROFIBUS layers (Data Link and Application Layers) and consequently the effort for protocol development is reduced. The support of repeaters requires a specific setting on some PROFIBUS timing parameters (the `Slot Time` and the `Idle Time`), which results in a lower responsiveness of the network to failures and on an increased latency of the message cycles. Additionally, the use of repeaters creates a broadcast network, so an error in a network domain will be propagated to the other domains.

An alternative approach is the use of Intermediate Systems behaving as bridges, thus operating at the Data Link Layer level. Therefore, the main research objectives of this thesis were the design of a hybrid wired/wireless PROFIBUS bridge-based architecture, in which bridge devices interconnect the wired and wireless domains of the network (this solution creates a potentially more reliable network at the cost of more complex intermediate systems and supporting protocols).

The hypothesis is that such architecture is devisable, while guaranteeing total compatibility with the existing PROFIBUS standard and coping with the original real-time capabilities of PROFIBUS. Moreover, we aimed at demonstrating that such an approach, in relation to RFieldbus, has a higher level of fault isolation, has increased responsiveness to errors and has better response times for transactions between network stations in the same network segment.

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 several important contributions to the development of a hybrid wired/wireless network architecture based on a standard fieldbus protocol – PROFIBUS.

The rest of this section summarises the main contributions of this thesis.

10.2.1. Hybrid Wired/Wireless PROFIBUS Bridge-based Architecture

This thesis proposed a network architecture which extends PROFIBUS in order to support wired and wireless stations in the same network.

In this proposal, the wired stations maintain total compatibility with legacy PROFIBUS stations. The wireless stations can be supported by the 802.11b Physical Layer (as in the RFieldbus project) and the remaining communication layers are compliant with the PROFIBUS standard. Therefore, a high level of compatibility with standard PROFIBUS is achieved, from the Application Layer perspective.

Transactions between stations in different domains (Inter-Domain Transactions (IDT)) are supported by bridge-like devices, which are capable of transparently relaying frames between the initiator and the responder. Each bridge is constituted by two modified PROFIBUS masters (the bridge masters), connecting the bridge to each domain. The bridge masters are responsible for handling IDTs and for supporting the mobility of mobile wireless stations and mobile wired domains.

Each wired/wireless domain has its own logical ring, which includes not only the domain masters, but also the bridge masters of the bridges connected to that domain. Therefore, a Multiple Logical Ring (MLR) network is created.

Chapter 4 also proposed an analytical model of the network, which can be used to describe the configuration and parameters of any network scenario. Additionally, this model is also used for the timing analysis carried out in Chapters 7 and 8.

10.2.2. Inter-Domain Protocol (IDP)

A consequence of the MLR approach is that when a master makes a request addressed to a station in another domain, it will not receive an immediate response from the responder. Therefore, Chapter 5 proposes a suitable protocol for handling such kind of transactions – the Inter-Domain Protocol (IDP).

The IDP explores some PROFIBUS-DP protocol features at the Data Link Layer (DLL) and Application Layer (AL), which enable a master to repeat the same request until receiving a response from the responder station. At the same time, the request is coded as an Inter-Domain Frame (IDF) by the bridges and relayed until reaching the responder. Then, the response is routed back to the first bridge master in the communication path (BM_{ini}) and stored. Since the initiator periodically repeats the same request, then, in a subsequent request, the BM_{ini} replies to the initiator with the previously stored response message.

In this way, neither the initiator nor the responder perceives that their transaction is being relayed by the bridges, leading to full transparency.

In summary, the IDP specifies the behaviour of the bridges and respective bridge masters, namely how the IDTs are handled, how proper routing is performed, how frames are codified and how errors are handled.

10.2.3. Inter-Domain Mobility Procedure (IDMP)

Chapter 6 proposes the Inter-Domain Mobility Procedure (IDMP), which enables stations to move between different wireless domains. The proposed IDMP is transparent to the system applications by guaranteeing that during its progress it does not generate any errors and there is no order inversion of frames. Additionally, the IDMP has been specified with the intent of being compatible with standard PROFIBUS stations.

The proposed mechanism relies on special functionalities supported by the system bridge masters. The Global Mobility Manager (GMM) functionality controls the evolution of the IDMP in the overall network. The Domain Mobility Manager (DMM) functionality controls certain phases of the procedure in its domain (like the `Beacon` transmission and the detection of stations entering the domain). The bridge masters are also required to stop accepting IDTs during the progress of the IDMP. Finally, the mobile wireless stations must be able to detect the transmission of `Beacons` and assess the quality of the radio channels in every network wireless domain. Additionally, the bridge masters are required to provide functionalities to detect the entrance of mobile wireless stations into the new domain and to update (through the transmission of specific protocol messages) the system routing tables accordingly.

10.2.4. Timing Analysis

A crucial factor on demonstrating the ability of the proposed hybrid wired/wireless PROFIBUS bridge-based network to support real-time applications is the provision of an analysis of its timing behaviour.

The time bounds presented for the IDP are based on existing timing analysis, for Single Logical Ring (SLR) PROFIBUS networks and on some contributions, described in Chapter 3. Chapter 7 proposes the formulations that permit the calculation of the worst-case response time for a message transaction in a PROFIBUS bridge-based network assuming that the IDMP is not active. Additionally Chapter 7 also analyses the latencies of the different phases of the IDMP. These results are then used in Chapter 8 to incorporate the effects of the IDMP on the formulation related to the worst-case response time of a message transaction.

10.2.5. Comparing with the RFieldbus Approach

In Chapter 9, a performance comparison between the proposed bridge-based approach and the repeater-based approach (RFieldbus) is performed. From that comparison, it has been shown that the bridge-based approach has a higher fault isolation level, since errors in one domain do not interfere with other message transactions which do not involve the transmission of frames on the error-prone domain.

In terms of responsiveness to errors it has been shown that on the repeater-based approach it was necessary to set the network with a much larger `Slot Time` parameter, and that this parameter is influenced by changes on the network bit rate, and frame size, contrarily to the bridge-based network. Additionally, the WCRT of the bridge-based approach is smaller that the WCRT of a repeater-based approach in what concerns Intra-Domain Transactions (IADT). Nevertheless, it has also been shown that the bridge-based approach implies larger WCRT for IDTs, mainly due to the effect of the IDMP.

10.3. Future work

The proposed architecture and mechanisms are considered to be adequate for the support of hybrid wired/wireless PROFIBUS bridge-based networks, and their main temporal characteristics can be adequately determined by the timing analysis methodologies proposed in this thesis.

However, the research work on hybrid wired/wireless fieldbus networks has not yet reached mature solutions to the problem, which results in a reduced number of commercial solutions with very limited functionalities and high cost.

This field of research can benefit from the evolution of radio technologies standardisation efforts (like the efforts being made by the IEEE 802 LAN/MAN Standards Committee), and from the research carried out in the adjacent scientific field of *Ad-hoc* and sensor networks. Additionally, many research efforts are also being carried out for the provision of low-power consuming devices and wireless transmission of energy. These advances may rule out the last cable from industrial communication infrastructures – the power cable, therefore providing a totally wireless solution.

In relation to the work presented in this thesis, some additional work and improvements which can still be made, are outlined next

The operation of the proposed architecture relies completely on the bridge devices, which are new components in a PROFIBUS network infrastructure. The bridge masters defined in this architecture are only able to independently send messages during the progress of the IDMP; that is, they do not have Application Layer functionalities implemented. An obvious add-on would be to include Application Layer functionalities in the bridges, as in a standard PROFIBUS-DP master. This would permit to improve the performance of the network, and at the same time reducing the number of devices required. Nevertheless, the inclusion of such functionality on the present framework would require the reformulation of the bridge architecture and corresponding timing analysis.

The proposed Inter-Domain Protocol is not the only possibility for the support of transactions between stations in different domains. Consequently, it would be interesting to address protocols with similar objectives and compare their behaviour and adapt them to the case where the bridge masters may also act as transaction initiators. Additionally, it would also be interesting to study if there are further possibilities for improving the performance of the IDP in respect to the coding of the frames and behaviour adopted by the bridge masters.

PROFIBUS-DP exists in several versions. The proposed architecture was designed assuming the basic functionalities of PROFIBUS-DP V0. Future work could focus on extending the capabilities of the architecture and protocols in order to support other technologies defined for PROFIBUS-DP, e.g. class 2 masters and the isochronous slave mode.

The design of the Inter-Domain Mobility Procedure is based on a set of assumptions which may be somewhat relaxed. As an example, if it is assumed that it is not necessary to execute the sequential transfer of data between stations, and that the loss of frames does not affect the applications, then some of the phases of the Inter-Domain Mobility Procedure could be simplified (or even eliminated). There are also some other possibilities to enhance the IDMP, like assuming other types of channel assessment mechanisms which would not require the emission of `Beacons` by the base stations or by increasing the priority of mobility related messages on the bridges, just to mention a few.

The timing analysis proposed in Chapters 7 and 8 has a high level of pessimism, essentially due to the fact of being based on worst-case assumptions. Therefore, it would be worthwhile to deeply investigate the sources of pessimism in order to try to reduce them. Additionally, a probabilistic timing analysis or a more thorough simulation-based study for the proposed architecture would also be a very an interesting development, especially for supporting soft real-time applications.

PROFIBUS International, the organisation that supervises the development of PROFIBUS standards, has recently proposed the PROFINET protocol standard, which is based on Ethernet. As in the case of PROFIBUS, PROFINET does not define any extensions for wireless support. Thus, it would also be worthwhile to investigate the possibilities of using a similar architecture for the support of hybrid wired/wireless communications in PROFINET networks, as also in other emerging fieldbus standards based on Ethernet.

References

Alves M., Tovar E., Vasques F., Hammer G., Röther K. (2002). Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-based Networks. In Proceedings of the 14th Euromicro Conference on Real-Time Systems (ECRTS'02), Vienna, Austria, pp. 142-150.

Alves M., (2003). Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-based Networks. PhD Dissertation, Universidade do Porto – Faculdade de Engenharia.

Behaeghel S., Nieuwenhuyse K., Marques L., Alves M., Tovar E. (2003). Engineering Hybrid Wired/Wireless Fieldbus Networks - a case study. In Proceedings of the 2nd International Workshop on Real-Time LANs in the Internet Age (RTLIA03), Porto, Portugal, pp. 111-114.

Cavalieri S., Panno D. (1997). On the Integration of Fieldbus Traffic within IEEE 802.11 Wireless LAN. In Proceedings of the 2nd IEEE International Workshop on Factory Communication Systems (WFCS'97), Barcelona, Spain, pp. 131-138.

Cavalieri S., Monforte S., Tovar E., Vasques F., (2002). Multi-Master Profibus-DP Modelling and Worst-Case Analysis Based Evaluation. In Proceedings of the 15th IFAC World Congress on Automatic Control, Barcelona, Spain.

El-Hoiydi A., Dallemagne P. (2000). Influence of Roaming on Real-Time Traffic in Wireless Networks. In Proceedings of the WIP Session of the 3rd IEEE International Workshop on Factory Communication Systems (WFCS'00), Porto, Portugal.

EN 50170 (1996). General Purpose Field Communication System. Volume 1 – P-NET, Volume 2 - PROFIBUS, Volume 3 – WorldFIP, European Norm.

Ferreira L., Alves M., Tovar E. (2002). Hybrid Wired/Wireless PROFIBUS Networks Supported by Bridges/Routers. In Proceedings of the 4th International Workshop on Factory Communication Systems (WFCS'02), Västerås, Sweden, pp. 193-202.

Ferreira L., Tovar E., Alves M. (2003a) PROFIBUS Protocol Extensions for Enabling Inter-Cell Mobility in Bridge-based Hybrid Wired/Wireless Networks. In Proceedings of the 5th IFAC International Conference on Fieldbus Systems and their Applications (FET03), Aveiro, Portugal, pp. 283-290.

Ferreira L., Tovar E., Alves M. (2003b). Enabling Inter-Domain Transactions in Bridge-Based Hybrid Wired/Wireless PROFIBUS Networks. In Proceedings of the 9th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA2003), Lisbon, Portugal, pp. 15-22.

Ferreira L., Tovar E. (2004a). Timing Analysis of an Inter-Cell Mobility Procedure for a Wired/Wireless PROFIBUS Network. In Proceeding of the 10th International Conference on Real-Time and Embedded Computing Systems and Application – RTCSA 2004, Gotemburg, Sweden, pp. 591-610.

Ferreira L., Tovar, E. (2004b). Timing Analysis of a Multiple Logical Ring PROFIBUS Network. In Proceeding of the 2004 IEEE International Workshop on Factory Communication Systems (WFCS'04), Vienna, Austria, pp. 81-90.

Fonseca J., Martins E., Almeida L., Pedreiras E. Neves P. (2000). Flexible Time-Triggered Protocol for CAN – New Scheduling and Dispatching Solutions. In Proceedings of IEEE International Conference on Communications 2000, New Orleans, USA.

Grow, R. (1982). A Timed Token Protocol for Local Area Networks. In Proc. of Electro'82, Token Access Protocols, Paper 17/3.

Hadzilacos, V. and Toueg, S. (1993). Fault-Tolerant Broadcasts and Related Problems. In Distributed Systems, Mullender, S. (Ed.), 2nd Ed., Addison-Wesley.

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 802.15.1 (2002). Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). LAN/MAN Standards Committee of the IEEE Computer Society, USA.

IEEE 802.15.4 (2003), Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). LAN/MAN Standards Committee of the IEEE Computer Society, USA.

IEC 61158-SER (2005), Digital data communications for measurement and control - Fieldbus for use in industrial control systems. International Electrotechnical Committee, Switzerland

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.

Lee K., Lee S. (2001). Integrated network of PROFIBUS-DP and IEEE 802.11 Wireless LAN with Hard Real-Time Requirement. In Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE'01), Pusan, Korea, pp. 1484-1489.

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.

Miorandi D., Vitturi S. (2004a). A Wireless Extension of PROFIBUS DP based on the Bluetooth System.In Journal of Computer Communications. Vol. 27, No 10, pp 946-960.

Miorandi D., Vitturi S. (2004b). Hybrid Wired/Wireless Implementations of PROFIBUS DP: a Feaseability Study Based on Ethernet and Bluetooth. To be published in *Ad-Hoc* Networks Journal.

Morel P. (1995). Mobility in MAP Networks using the DECT Wireless Protocols. In Proceedings of the 1st Workshop on Factory Communication Systems, Leysin, Switzerland, pp. 145-149.

Morel, P., Muralt, R., and Decotignie, J.-D. (1995). A Wireless Extension for Fieldbus. In Proceedings of the 2nd International Conference on Industrial Automation, Nancy, France, pp. 275-279.

Pacheco F., Tovar E. (2002). User-interface Technologies for the Industrial Environment: Towards the Cyber-factory. In Proceedings of the 6th CaberNet Radicals Workshop, Funchal, Madeira Island, February 2002.

Rauchhaupt, Lutz (2003). RFieldbus a Survey. In Proceedings of the 5th IFAC International Conference on Fieldbus and their Applications, Aveiro, Portugal.

Roberts, D.A. (1993). OLCHFA a Distributed Time-Critical Fieldbus. In Proc. of IEE Colloquium in Safety Critical Distributed Systems, London, United Kingdom, pp. 6/1-6/3.

Sousa, P., Ferreira, L. (2005) Hybrid Wired/Wireless PROFIBUS Network Simulator. Technical-Report Hurray-tr-050402. http://www.hurray.isep.ipp.pt. Polytechnic Institute of Porto.

Stankovic J. (1988). Real-Time Computing Systems: the Next Generation. In *Tutorial: Hard Real-Time Systems*, Stankovic, J. and Ramamritham (Editors), IEEE Computer Society Press, Los Alamitos, USA, pp 14-38.

Tovar E, Vasques F. (1999a). Real-Time Fieldbus Communications Using PROFIBUS Networks. In IEEE Transactions on Industrial Electronics, Vol. 46, No 6, pp. 1241-1251.

Tovar E. Vasques F. (1999b). Cycle Time Properties of the PROFIBUS Timed Token Protocol. In Computer Communications, Elsevier Science, No 22, pp. 1206-1216.

Tovar E., Pinho M., Alves M., Pacheco F. (2003). Bringing Industrial Multimedia to the Factory-Floor: What is at stake with RFieldbus. In Proceedings of the 5th IFAC International Conference on Fieldbus Systems and their Applications (FET'03), Aveiro, Portugal, July, pp 131-138.

Varga A. (2005). OMNet ++ User Manual, version 3.1. Available at http://www.omnetpp.org/doc/manual/usman.html.

Willig A. (1999). Analysis of the PROFIBUS Token Passing Protocol over Error Prone Links. In Proceedings 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.

Willig A, (2003). Polling-Based MAC Protocols for Improving Real-Time Performance in a Wireless PROFIBUS. In IEEE Transactions on Industrial Electronics, Vol. 50, No 4, August, pp. 806-817.

ZigBee, (2004). ZigBee Specification. ZigBee Alliance.

Annex

Acronyms and Symbols

This annex presents two list one containing the acronyms and another containing the symbols used in this thesis.

1. Acronyms

Table A.1 – List of acronyms

Acronym	Description	Section
ADDR	Station address	4.6.2
AGV	Automatic Guided Vehicle	1.1
AL	Application Layer	2.2.1
AWlD	*Ad-hoc* Wireless Domain	4.2
BER	Bit Error Rate	2.4
BM	Bridge Master	4.3
BM_{ini}	First bridge master in the path from the initiator to the responder of a transaction	4.4.1
BM_{res}	Last bridge master in the path from the initiator to the responder of a transaction	4.4.1
BS	Base Station	1.2
BT	Beacon Trigger	1.2
CAN	Controller Area Network	2.4
COTS	Commercial-Off-The-Shelf	1.2
CSRD	Cyclic Send and Receive with Data (PROFIBUS standard)	2.2.2
DA	`Destination Address` (PROFIBUS standard)	2.2.2
DAE	`Destination Address Extension` (PROFIBUS standard)	2.2.2
DCCS	Distributed Computer-Controlled System	1.1
DECT	Digital Enhanced Cordless Telecommunications	2.4
DLL	Data Link Layer	1.3
DMM	Domain Mobility Manager	4.4.2
DRWlM	Domain Resident Wireless Master Station	4.2.1
DRWlS	Domain Resident Wireless Slave Station	4.2.1
DSSS	Direct Sequence Spread Spectrum	2.3.1
EFC	`Embedded frame Function Code`	5.3.4
EFT	`Embedded Frame Type`	5.3.4
ES	End System	4.2
ETSI	Institut Européen des Normes de Télécommunication	2.4
FC	`Frame Control` (PROFIBUS standard)	2.2.2

FCS	Frame Check Sequence (PROFIBUS standard)	2.2.2
FMA1/2	Management for PROFIBUS networks layers 1 and 2	2.2.1
GAPL	Gap List	2.2.2
GMM	Global Mobility Manager	4.4.2
I/O	Input/Output	2.2.1
IADT	Intra-Domain Transaction	4.4.1
IDF	Inter-Domain Frame	4.4.1
IDMP	Inter-Domain Mobility Procedure	1.4
IDP	Inter-Domain Protocol	1.4
IDreq	Inter-Domain Request frame	4.4.1
IDres	Inter-Domain Response frame	4.4.1
IDT	Inter-Domain Transaction	4.1
IEC	International Electrotechnical Commission	2.2.3
IS	Intermediate System	1.3
ISA	The Instrumentation, Systems, and Automation Society	2.4
ISO	International Organization for Standardisation	2.2.1
LAN	Local Area Network	2.4
LAS	List of Active Stations (PROFIBUS Standard)	2.2.2
LE	Frame Length (PROFIBUS Standard)	2.2.2
LEr	Frame Length repeated (PROFIBUS Standard)	2.2.2
LIS	Link Intermediate System	2.3
LL	Live List (PROFIBUS Standard)	2.2.2
LOT	List of Open Transactions	5.2
MAC	Medium Access Control	1.2
MAP	Manufacturing Automation Protocol	2.4
MLIS	Mobile Linking Intermediate System	4.2.2
MLR	Multiple Logical Ring	1.3
MMS	Manufacturing Message Specification	2.4
MOFDI	Mobile Fieldbus Devices in Industry	2.4
MS	Mobile Wireless Slave Station functionality	4.5.3
MWlM	Mobile Wireless Master Station	4.2.1
MWlS	Mobile Wireless Slave station	4.2.1
MWrD	Mobile Wired Domain	4.2.2
NS	Next Station (PROFIBUS Standard)	2.2.1
OLCHFA	Open Low-Cost Time-Critical Wireless Fieldbus Architecture	2.4
OSI	Open System Interconnection	2.2.1
PBT	Prepare_for_Beacon_Transmission	6.2.3
PC	Personal Computer	2.2.1
PCF	Point Coordinator Function (IEEE 802.11)	2.4
PDA	Personal Digital Assistant	1.1
PDU	Protocol Data Unit	4.6.1
PhL	Physical Layer	1.2
PLC	Programmable Logical Controller	10.3
PROFIBUS	PROcess FIeld BUS	1.2
PROFIBUS-DP	PROFIBUS – Decentralised Peripherals	2.2.2
PROFIBUS-FMS	PROFIBUS – Fieldbus Message Specification	2.2.2
PS	Previous Station (PROFIBUS Standard)	2.2.1

RBT	Ready_for_Beacon_Transmission	6.2.3
RFieldbus	High Performance Wireless Fieldbus in Industrial Multimedia-Related Environment	1.2
RSMP	Ready_to_Start_Mobility_Procedure	6.2.3
RT	Routing Table	5.2.2
RU	Route_Update	6.2.3
SA	Source Address (PROFIBUS standard)	2.2.2
SAE	Source Address Extension (PROFIBUS standard)	2.2.2
SBT	Start_Beacon_Transmission	6.2.3
SC	Short Acknowledge (PROFIBUS standard)	2.2.2
SD	Start Delimiter (PROFIBUS standard)	2.2.2
SDA	Send Data with Acknowledge (PROFIBUS Standard)	2.2.2
SDN	Send Data with no Acknowledge (PROFIBUS Standard)	2.2.2
SIS	Structuring Intermediate System	4.2.2
SLIS	Structuring & Linking Intermediate System	4.2.2
SLR	Single Logical Ring	2.2.2
SMP	Start_Mobility_Procedure	6.2.3
SRD	Send and Request Data (PROFIBUS Standard)	2.2.2
SWlD	Structured Wireless Domain	4.2
TCP/IP	Transport Control Protocol/Internet Protocol	6.2.1
TDMA	Time Division Multiple Access	2.4
TI	Transaction Identifier	5.2.3
TS	This Station (PROFIBUS Standard)	2.2.2
WCRT	Worst-Case Response Time	1.4
WlBM	Wireless Bridge Master	4.3
WlM	Wireless Master Station	4.2.1
WlS	Wireless Slave Station	4.2.1
WorldFIP	Factory Instrumentation Protocol	2.4
WrBM	Wired Bridge Master	4.3
WrM	Wired Master Station	4.2.1
WrS	Wired Slave Station	4.2.1

2. Symbols

Table A.2 – List of symbols

Acronym	Description	Section
$\beta_{tc}^{k_i}$	Represents if a FDL_Request_Status is transmitted during token cycle *tc* or not.	3.3.2
ϕ	Bridge internal forwarding delay	4.5.4
λ	Total token latency	3.2
$\Delta h_{tc}^{k_i}$	Value of the token holding timer at the token arrival to master k_i in its tc^{th} token cycle	3.2

$\Delta l_{tc}^{k_i}$	Time available for processing low-priority messages at the tc^{th} token cycle	3.2
$(C_{Inq})^{dmm}$	Worst-case latency of the `Inquiry` request message on a domain (represented by its DMM)	7.3.2
$(C_{Ires})^{dmm}$	Worst-case latency of the `Inquiry` response message on a domain (represented by its DMM)	7.3.2
$(Creq_i^k)^d$	Latencies associated to the transmission of a request, from message stream *i* from master *k*, on a network domain *d*	7.2.1
$(Cresp_i^k)^d$	Latencies associated to the transmission of a response, from message stream *i* from master *k*, on a network domain *d*	7.2.1
Π_{master}	Set of mobile wireless masters in the system	7.3.4
$(n_{iBMs})^{dmm}$	Number of BMs which are inquired by the DMM during the inquiry sub-phase	7.3.2
Π_{slave}	Set of mobile wireless slaves in the system	7.3.4
$(t_{IDT\_MMM\_dis}^{k})^{orig\rightarrow dest}$	Worst-case time span during which master *k* has its IDTs disabled when moving between the original domain *org* and the destination domain *dest*.	8.3.3
A_i^k	Number of retries executed by the initiator, in the case of an IDT, before obtaining a response from BM_{ini}, related to message stream *i* from master *k*	7.2.1
b	Number of bridges between the initiator and the responder of an IDT	7.2.1
B^k	Initial blocking caused by other masters with message transactions already going	3.2
C	Total duration of a message cycle	3.2
C_{disc}^{dmm}	Worst-case latency associated with the `Discovery` message on a domain represented its *dmm*, including the request and response frames	7.3.4
C_{FDL}	Worst-case latency of the `FDL_Request_Status` message and respective response	3.3.2
C_{FDL_req}	Worst-case latency of the `FDL_Request_Status` request message	3.3.2
C_{FDL_res}	Worst-case latency of the `FDL_Request_Status` response message	3.3.2
Ch_i^k	Worst-case duration for a high-priority message cycle related to message stream *i* from master *k*	3.2
$Cl_{\max}^{k_i}$	Longest low-priority message cycle performed by master k_i	3.2
$Ch_{\max}^{k_i}$	Longest high-priority message cycle performed by master k_i	3.2

C_i^k	Worst-case duration for a message cycle related to message stream *i* from master *k*	4.6
C_{req}	Duration of a request PDU	2.3.1
C_{resp}	Duration of a response PDU	2.3.1
C_σ	Longest message cycle in a single logical ring network	3.2
d	bits per char for the PhL protocol	4.5.1
D	Set of communication domains	4.5.1
D_PPAR	Set of PROFIBUS parameters which are common to all master stations in a domain	4.5.1
D_TYPE	Represents the Communication Domain's type; *D_TYPE* ∈ {WrD, SWlD, AWlD, MWrD}	4.5.1
D^i	Communication Domain *i*	4.5.1
$D_{k \to i}$	Distance parameter	3.3.2
G	Gap Update Factor	3.3.2
$H_{tc}^{k_i}$	Time that master k_i spent processing message cycles during the tc^{th} token visit, note that this value can be higher then $\Delta h_{tc}^{k_j}$	3.2
HSA	Highest Station Address	2.2.2
I^k	interference caused by high-priority message streams (from master k and the other masters) and low-priority message streams (from other masters)	3.2
IS	Set of Intermediate Systems in the network	4.5
$IS(D^i, t)$	Function that returns the set of all ISs that are associated to D^i at time *t*.	4.5.1
IS_TYPE	Type of IS; *IS_TYPE* ∈ {LIS, SIS, SLIS, MLIS}	4.5.4
IS^i	Intermediate system *i*	4.5
k_i	Used to designate master *i*	3.2
L	Number of characters in a frame	4.5.1
lH^i	Overhead of the frame head per PhL PDU in physical medium i	4.5.1
L_{req}	Length of the PROFIBUS standard DLL request message	4.6
L_{resp}	Length of the PROFIBUS standard DLL response message.	4.6
lT^i	Overhead of the frame tail per PhL PDU in physical medium *i*	4.5.1
M	Set of master stations in the network	4.5
$M(D^i, t)$	Function that returns the set of all masters that are associated to D^i at time *t.*	4.5.1
M_PPAR	Set of PROFIBUS parameters related to a master station	4.5.2
$\max T_{SDR}$	Maximum delay before a responder starts transmitting a response to a request.	4.5.2

max_retry_limit	maximum number of retries, performed by the PROFIBUS DLL, before giving up a request	2.2.2
M^i *or Mi*	Master *i*	4.5.2
min T_{SDR}	Minimum delay before a responder starts transmitting a response to a request.	4.5.2
MOB_FUNCT	Mobility functionalities which are supported by a station; in the case of a master: *MOB_FUNCT* ∈ {MS, DMM, GMM, BR}; in the case of a slave: *MOB_FUNCT* ∈ {MS}	4.5.2
n	Designates the number of master stations which belong to a logical ring	3.2
$n_{beacons}$	Number of beacons transmitted by a SIS (base station) or by a bridge of the type SLIS.	4.5.4
nh^k	Number of high-priority messages which can be simultaneously queued in master *k*	3.2
$nh\pi_{tc}{}^{k}$	Number of high-priority message cycles processed by master k_i in its tc^{th} token cycle	3.2
nl^k	Number of low-priority messages which can be simultaneously queued in master *k*	4.5.2
$nl\pi_{tc}^{k_i}$	Number of low-priority message cycles processed by master k_i in its tc^{th} token cycle	3.2
$n_{mob\_stations}$	Number of mobile wireless stations (including masters and slaves) in a system	7.3.4
P_i^k	Priority for message stream *i* from master k	4.6
Q^k	Queuing delay in master *k*	3.2
$Rbmi_i^k$	This timing designates, in the case of an IDT, the time that elapses from the reception of the request at the BM_{ini}, until the arrival of the respective response	7.2.1
$Rbmi\_m_i^k$	The same as $Rbmi_i^k$, but considering the influence of the IDMP	8.3.2
r^i	Bit rate in physical medium i	4.5.1
$Rinq_{RBT}^{x\rightarrow(x+1)}$	delay encountered by the `Ready_for_Beacon_Transmission` message when being transmitted from a BM *x* to another BM *x+1*, during the inquiry subphase	7.3.2
$Rmlr_i^k$	Worst-case response time on a multiple logical ring PROFIBUS network	7.2.1
$Rmlr\_m_i^k$	Worst-case response time, including the latencies of the IDMP, on a multiple logical ring PROFIBUS network	8.3.2
$Rslr_i^k$	Worst-case response time for message stream *i* from master *k* in a single logical ring configuration or for an IADT	3.2

$Rslr\_m_i^k$	Worst-case response time, including the latencies of the IDMP, for message stream *i* from master *k*, which is an IADT	8.3.1
$Ru_i^{k\rightarrow w}$	Worst-case time required by a request, from a message stream *i*, to go from a master *k* to another station *w*, using unicast transmission mode	7.2.2
S	Set of slaves in the network	4.5
$S(D^i, t)$	Function that returns the set of all slaves that are associated to D^i at time *t*	4.5.1
S_PPAR	Set of PROFIBUS parameters for a slave station	4.5.3
S^i *or Si*	Slave *i*	4.5.3
S_i^k	Message stream *i* from master *k*	4.6
ST_TYPE	In the case of master: station's type; $ST\_TYPE \in$ {WrM, RWlM, MWlM, WrBM, WlBM} In the case of a slave: station's type; $ST\_TYPE \in$ {WrS, RWlS, MWlS}	4.5.2
St^i	Message stream set. This set can only be associated with WrM, DRWlM, MWlM	4.5.2
t_{beacon}^{dmm}	Worst-case latency associated with the transmission of a beacon frame	7.3.3
t_{BT}	Time required for the transmission of the `Beacon Trigger` message on the repeater-based approach	3.3.3
tc	Refers to the tcth token visit to a master	3.2
$t_{cap\_token}^{dmm}$	Worst-case time required until capturing the token	7.3.2
$T_{cycle\_mob}^{k}$	Duration of the token cycle time during the execution of the mobility procedure on an Repeater-based network	3.3.3
T_{cycle}^{k}	Worst-case token rotation time related to master *k* or in a specific network domain	3.2
t_{disc}^{dmm}	Worst-case duration of the station discovery sub-phase	7.3.4
T_{EH}	Value for the *IDT Error Handling Timer*	5.3.5
$t_{fin\_IDT}^{bm}$	Worst-case time until all IDTs are completed for a particular BM bm	7.2.1
T_{GUD}	`Gap Update` time, defines the periodicity of the GAP update mechanism	2.2.2
$t_{IADT\_dis\_wr}^{dmm}$	Worst-case time during which IADT are disabled in a wired domain (represented by its *dmm*)	8.2.1
$t_{IADT\_dis\_wr}^{dmm}$	Worst-case time during which IADT are disabled in a wireless domain (represented by its *dmm*)	8.2.1
T_{ID1}	Idle time inserted by a master station after an acknowledgement, response or token PDU	2.2.2

T_{ID2}	Idle time inserted by a master ES after an acknowledged request PDU (PROFIBUS).	2.2.2
$t_{IDT\_dis\_wl}^{bm}$	Worst-case time during which IDTs are disabled in a wireless bridge master	8.2.2
$t_{IDT\_dis\_wr}^{bm}$	Worst-case time during which IDTs are disabled in a wired bridge master	8.2.2
$t_{IDT\_MM\_dis}^{bm,s}$	Worst-case time, during which IDTs related to a mobile wireless masters *s*, are disabled in a BM *bm*	8.2.3
$t_{IDT\_mob\_dis}^{BMini,s}$	Worst-case time during which IDTs involving a mobile wireless station *s* are disabled, on its BM_{ini}, from the reception of the `Start_Mobility_Procedure` message until the reception of a `Route_Update` message regarding that station	8.3.3
$t_{IDT\_MS\_dis}^{bm,s}$	Worst-case time, during which IDTs related to a mobile wireless slave *s*, are disabled in a BM *bm*	8.2.3
T_i^k	Minimum message stream period	4.6
$t_{ina}^{s,dmm\rightarrow dmm'}$	Worst-case time during which a mobile wireless station *s* is inaccessible, when it moves from the wireless domain of *dmm* to the wireless domain of *dmm'*	8.3.4
$t_{master\_entry}^{k,j}$	Worst-case time required by a master *j* to enter into a PROFIBUS logical ring, in the GAP interval controlled by master *k*	3.3.2
t_{mob}	Worst-case duration of the inter-domain mobility procedure	7.3.5
t_{PBP}^{dmm}	worst-case time required for the `Prepare_for_Beacon_Phase` message to reach DMM *dmm*	7.3.2
t_{per_mob}	Periodicity of the mobility procedure in the repeater-based approach	3.3.3
t_{phase1}	Worst-case duration of the Inter-Domain Mobility Procedure Phase 1	7.3.1
t_{phase2}	Worst-case duration of the Inter-Domain Mobility Procedure Phase 2	7.3.2
t_{phase3}^{dmm}	Worst-case duration of Phase 3, calculated for a wireless domain *d* (represented by its DMM)	7.3.3
$t_{phase4}^{s,bm}$	Worst-case duration of phase 4, which is equivalent to the time required before a BM *bm* knows that a station *s* is again operational in a wireless domain	7.3.4
T_{QUI}	Transmitter fall time	4.5.2
t_{RBP}^{dmm}	Worst-case time required by the `Ready_for_Beacon_Phase` message to go from the DMM dmm until the GMM	7.3.2

T_{RDY}	Time within which a master station shall be ready to receive an acknowledgement or response after transmitting a request.	4.5.2
$Treq_i^k$	Time required to transmit a request frame associated with message stream *i* from master *k*	4.6
$Tres_i^k$	time required to transmit a response frame associated with message stream *i* from master *k*	4.6
t_{RSMP}^{bm}	Worst-case time required by the Ready_to_Start_Mobility_Procedure message to go from a BM *bm* to the system GMM	7.2.1
$t_{RU,s}^{bm}$	Worst-case time span that the Route_Update message, relative to station *s*, needs to go from DMM *dmm* to a BM *bm*	7.3.4
t_{SB}^{dmm}	Worst-case time required by the Start_Beacon_Transmission message to reach a DMM *dmm*	7.3.3
T_{SDI}	Station delay of the initiator, which is measured with respect to the receipt of the last frame last bit until an initiator is ready to transmit again.	4.5.2
T_{SDI}	Station Delay of the Initiator	2.2.2
T_{SDR}	Station Delay of a Responder	2.2.2
T_{SET}	Set-up time which expires from the occurrence of an event (e.g. interrupt: last octet sent or Synchronous Time expired) until the necessary reaction is performed (e.g. to start Synchronous Time or to enable the receiver).	4.5.2
T_{SL}	The Slot Time is a parameter set in every master that defines the timeout for listening for activity in the bus, after having transmitted an acknowledged request or token.	2.2.2
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	2.2.2
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 station that received the token	2.2.2
T_{SM}	Safety margin (PROFIBUS)	2.2.2
t_{SMP}^{bm}	Worst-case time required by the Start_Mobility_Procedure message to reach a BM bm	7.3.1
t_{st}	System turnaround time	2.3.1

t_{st_token}	System turnaround time calculated for every master in the network after the transmission of a token frame	2.3.1
T_{SYN}	Synchronisation period of (at least) 33 idle bit periods	2.2.2
T_{TD}	Transmission Delay is the propagation delay in the bus.	2.2.2
T_{To}	Time-out timer	2.2.2
T_{TR}	Target Token Rotation time	2.2.2
w^i	Physical medium of D^{i}	4.6.1
Φ	Bridge internal relaying delay; Only applies to IS of the type: {SIS, SLIS}	4.5.4
$\Omega_{req}(S_i^k)$	IDTreq Communication path for message stream S_i^k	4.6
$\Omega_{res}(S_i^k)$	IDTres Communication path for message stream S_i^k	4.6

List of Publications Related to This Thesis

Tovar, E., Alves, M., Pacheco, F., Ferreira, L., Pereira, N., Machado, S. A High Performance Wireless Fieldbus in Industrial Multimedia-Related Environment. In Proceedings of the 4th CaberNet Plenary Workshop, Pisa, Italy, October 2001.

Ferreira, L., Machado, S., Tovar, E. Scheduling IP Traffic in Multimedia Enabled PROFIBUS Networks. In Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2001), Antibes - Juan les Pins, France, pp. 169-176, October 2001.

Tovar, E., Vasques, F., Ferreira, L., Pacheco, F. Industrial Multimedia over Factory-Floor Networks. In Proceedings of the 10th IFAC Symposium on Information Control Problems in Manufacturing (INCOM '01), Vienna, Austria, September 2001.

Ferreira, L., Tovar, E. QoS of IP Services in a Fieldbus Network: on the Limitations and Possible Improvements. In Wip Proceedings of the 14th Euromicro Conference on Real-Time Systems, Vienna, Austria, June 2002.

Ferreira L., Alves M., Tovar E. Hybrid Wired/Wireless PROFIBUS Networks Supported by Bridges/Routers. In Proceedings of the 4th International Workshop on Factory Communication Systems (WFCS'02), Västerås, Sweden, pp. 193-202, August 2002.

Ferreira L., Tovar E., Alves M. PROFIBUS Protocol Extensions for Enabling Inter-Cell Mobility in Bridge-based Hybrid Wired/Wireless Networks. In Proceedings of the 5th IFAC International Conference on Fieldbus Systems and their Applications (FET03), Aveiro, Portugal, pp. 283-290, July 2003.

Ferreira L., Tovar E., Alves M. Enabling Inter-Domain Transactions in Bridge-Based Hybrid Wired/Wireless PROFIBUS Networks. In Proceedings of the 9th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA2003), Lisbon, Portugal, pp. 15-22, September 2003.

Ferreira L., Tovar E. Timing Analysis of an Inter-Cell Mobility Procedure for a Wired/Wireless PROFIBUS Network. In Proceeding of the 10th International Conference on Real-Time and Embedded Computing Systems and Application – RTCSA 2004, Gothenburg, Sweden, pp. 591-610, September 2004.

Ferreira L., Tovar, E. Timing Analysis of a Multiple Logical Ring PROFIBUS Network. In Proceeding of the 2004 IEEE International Workshop on Factory Communication Systems (WFCS'04), Vienna, Austria, pp. 81-90, September 2004.

Ferreira L., Tovar, E. The Influence of Inter-Domain Mobility on Message Stream Response Time in Wired/Wireless PROFIBUS-based Networks. To appear in Proceeding of 5th IFAC International Conference on Fieldbus Systems and their Applications (FET'05), Puebla, Mexico, November 2005.