

## Research Article

# Hybrid Wired/Wireless PROFIBUS Architectures: Performance Study Based on Simulation Models

**Paulo Baltarejo Sousa and Luís Lino Ferreira**

*CISTER Research Group, School of Engineering, Polytechnic Institute of Porto, R. Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal*

Correspondence should be addressed to Paulo Baltarejo Sousa, pbs@isep.ipp.pt and Luís Lino Ferreira, llf@isep.ipp.pt

Received 8 June 2009; Revised 9 October 2009; Accepted 14 January 2010

Academic Editor: Christian Ibars

Copyright © 2010 P. B. Sousa and L. L. Ferreira. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The problem of providing a hybrid wired/wireless communications for factory automation systems is still an open issue, notwithstanding the fact that already there are some solutions. This paper describes the role of simulation tools on the validation and performance analysis of two wireless extensions for the PROFIBUS protocol. In one of them, the Intermediate Systems, which connect wired and wireless network segments, operate as repeaters. In the other one the Intermediate Systems operate as bridge. We also describe how the analytical analysis proposed for these kinds of networks can be used for the setting of some network parameters and for the guaranteeing real-time behaviour of the system. Additionally, we also compare the bridge-based solution simulation results with the analytical results.

## 1. Introduction

The rapid evolution of wireless technologies has made possible to extend its use into the factory automation field. Such solution benefits from several advantages in relations to their wired counterparts, namely, equipment installation which is easier, configuration flexibility, ability to evolve, and cuts in cabling and maintenance costs, just to mention a few.

Nevertheless, industrial automation systems usually have a different set of requirements in relation to most wireless applications; particularly, these networks must have a predictable timing behaviour, usually referred as real-time, and they must also be highly reliable. Real-time behaviour is usually assured by a proper Medium Access Control (MAC) mechanism, whose real-time behaviour must be guaranteed by an analytical analysis. Reliability is assured by the use of proper protocols which provide error detection and recovery with minimal interference on the system behaviour.

But, most of the industrial community is very reluctant to integrate new technologies in their consolidated automation systems, either by preconception or due to the immaturity of these technologies. When addressing communication

systems for control applications, these fears become even more acute. That is why only a few fieldbus communication systems consolidated their market position, due to their technical features and also to big enterprise lobbies. From these, PROFIBUS (PROcess FIEldBUS) [1] is the world's most successful fieldbus, with more than 25 million devices installed at the end of 2008 [2], and it continues to grow at a very interesting rate.

The RFieldbus architecture [3], driven by the European Project IST-1999-11316 consortium, has provided a complete solution where multiple segments and multiple wireless cells (hereafter, segments and wireless cells are referred as domains) are interconnected via Physical Layer (PhL) Intermediate Systems (operating as repeaters). This solution (validated by two field trials, one of them developed in our facilities [4]) is compatible with standard PROFIBUS, but the fact that all messages are broadcast throughout the network leads to some problems, namely, no error containment between different domains and low responsiveness to failures. These facts triggered the analysis and proposal of an alternative approach where the Intermediate Systems (ISs) operate at the Data Link Layer (DLL) level as bridges

[5, 6]. This approach requires two new protocols, one for supporting the communication between stations in different domains the Interdomain Protocol (IDP), and another to support the mobility of wireless stations between different wireless domains, the Interdomain Mobility Procedure (IDMP).

Although the bridge-based approach brings up some additional complexity, it showed up to overcome some limitations of the RFieldbus approach [7].

In this paper, we show how the simulation tools developed by us helped on the validation of the proposed protocol extensions. Additionally, we compare the simulation results with our analytical formulations, which proved the real-time behaviour of the network.

This paper is structured as follows. Section 2 presents the state-of-the-art on wireless fieldbuses. Section 3 presents the basics on the PROFIBUS and on the proposed solutions to extend it, one based on repeaters and the other based in bridges. We clearly give more emphasis to the bridge-based solution which has been the primary focus of our work. Section 4 describes the main building blocks of our simulator architecture, which has been implemented using the OMNeT++ framework. It is possible to prove and characterise the real-time behaviour of the network by providing a worst-case timing analysis, which is described in Section 5. Section 6 describes in detail a network scenario, whose results are presented and discussed in Section 7. Finally, in Section 8 we draw some conclusions.

## 2. Related Work

Industrial communications have progressed enormously in the last decade by replacing the traditional one-to-one connections between sensors/actuators and controllers with networked connections. This resulted in lower installation wiring costs and improved functionalities. But for some applications cables might not be an option, and wiring can be difficult and prone to problems. Examples of these kinds of applications are automation in harsh environments, communications with mobile robots, and communications with sensors and actuators in rotating machinery. Obviously, these applications collect enormous benefits from the adoption of wireless communication technologies.

Some solutions for providing traditional PROFIBUS with wireless extensions have been proposed [8–10]. Nevertheless, these solutions, which basically operate as gateways, are quite limited either in terms of number of segments/wireless cells and in the support of mobility.

More recently two commercial approaches have been proposed, which provide solutions to integrate wired PROFIBUS systems with wireless devices—Wireless Interface for Sensors and Actuators (WISA) and Wireless Hart.

WISA [11] is based on the IEEE 802.15.1 Physical Layer and on a Time Division Multiple Access (TDMA) MAC layer, which supports up to 120 devices in a star topology and has communication delays in the order of a few millisecond.

Wireless Hart [12] has been recently proposed, as a wireless extension of the Hart protocol, which is particularly

suited for process industries. This solution is based on the IEEE 802.15.4 PhL and on also on a TDMA MAC. Additionally, it provides multihop capabilities, which allows extending the network to wider areas.

PROFINET [13] also provides wireless capabilities based on the IEEE 802.11 protocol [14] plus industrial extensions for real-time and mobility. The real-time behaviour of the network is assured by using the Industrial Point Coordinator Function (iPCF) protocol, which permits the reservation of bandwidth between the stations in the network. Mobility is provided by existing extensions to 802.11. Connections to PROFIBUS networks can only be provided by PROFINET gateways.

Contrarily, to the previous approaches, that join new protocols to the PROFIBUS by means of specific purpose gateway devices, our proposal integrates wireless technology with PROFIBUS by interconnecting both networks with bridge-like intermediate systems.

This solution reduces the communication delays since the transaction initiators have direct access to the responding station, contrarily to the other solutions, which require a gateway to periodically acquire the data from the stations, store it internally, and wait for a request from the initiator.

Additionally, all of the approaches referred above do not support inter-cell mobility, except for PROFINET. Our solution uses the existing PROFIBUS mechanisms for network initialization and fault correction to support the mobility of nodes and most importantly it does so providing real-time guarantees.

Wireless PROFINET also provides mobility functionalities, but the time required for the handoff procedure is not known to be bound.

Traditionally, the PhL of PROFIBUS is wired; so in the bridge-based approach we assume that wireless stations can use any of the commonly available PhL, like IEEE 802.11b, IEEE 802.15.1, or IEEE 802.14.4. The layers above do not require any changes, except in the case of the bridge devices.

## 3. Hybrid Wired/Wireless Profibus Architectures

*3.1. Basics on Profibus.* The PROFIBUS Data Link Layer (DLL) uses a token passing procedure to grant bus access to masters, where the token is passed between masters in ascending Medium Access Control (MAC) address order, organizing the medium access in a logical ring.

After receiving the token, a PROFIBUS master is capable of dispatching transactions during its Token Holding Time ( $T_{TH}$ ), which is, for each token visit, the value corresponding to the difference, if positive, between the Target Rotation Time ( $T_{TR}$ ) parameter and the Real Rotation Time ( $T_{RR}$ ) of the token.

A transaction (or message cycle) consists in the request frame from the initiator (a master) and of the associated acknowledgement or response frame from the responder station. The acknowledgement (or response) must arrive before the expiration of the Slot Time ( $T_{SL}$ ); otherwise the initiator repeats the request a number of times defined

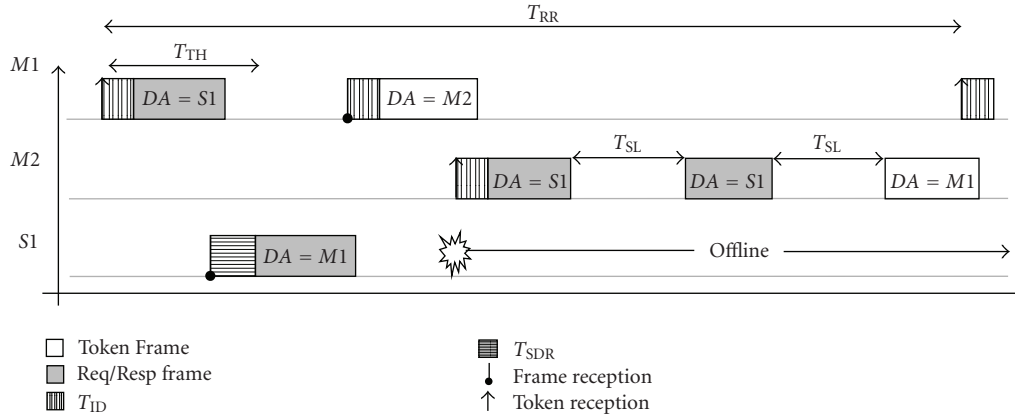


FIGURE 1: PROFIBUS message cycle timings.

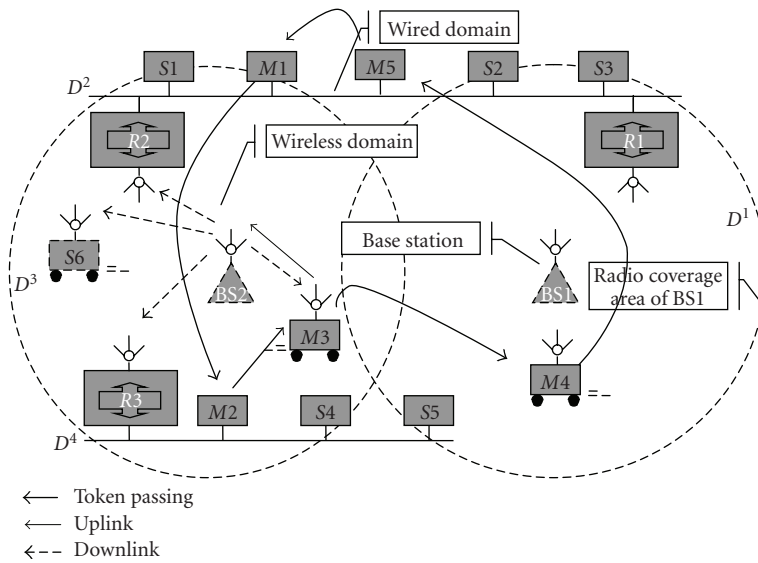


FIGURE 2: Repeater-based approach network scenario.

by an internal DLL variable called `max_retry_limit`. The Station Delay of Responder Time ( $T_{SDR}$ ) is the time required by a responder before transmitting a reply frame.

Idle Times ( $T_{ID1}$  and  $T_{ID2}$ ) are periods of inactivity inserted by master stations between two consecutive message cycles.  $T_{ID1}$  is inserted after an acknowledgement, response, or token and  $T_{ID2}$  after an unacknowledged request frame. In the remaining of this paper we will refer simply to Idle Time or  $T_{ID}$  in both situations. Figure 1 illustrates the previous concepts.

In order to detect the token lost is used a timer called Time-Out Time ( $T_{TO}$ ). A token lost is detected when a master does not detect any network activity for a time period defined by its  $T_{TO}$  timer, which is set to as follows:  $T_{TO} = 6 \times T_{SL} + 2 \times n \times T_{SL}$ , where  $n$  is the master address. This setting ensures that the master station with lowest address is the first to detect the token lost. After which, this station initiates a token recovery mechanism; for details see [1, 15, 16].

PROFIBUS also supports message transactions where just one message is transmitted between a master and a slave. These transactions is based on the Send Data with No acknowledge (SDN) service. Transaction involving a response and reply are based on the Send and Request Data (SRD) service.

**3.2. Repeater-Based Architecture.** In the repeater-based approach (Figure 2) the wireless stations are standard PROFIBUS stations, using a modified Physical Layer (PhL). In such a network, all messages are relayed through Base Stations (BSs) which operate in cut-through mode as a wireless repeaters, using two radio channels, one to receive frames from wireless stations (the uplink channel), and another to transmit frames to wireless stations (the downlink channel).

In the repeater-based approach, the repeaters receive frames from the wired domain, modify their PhL frame

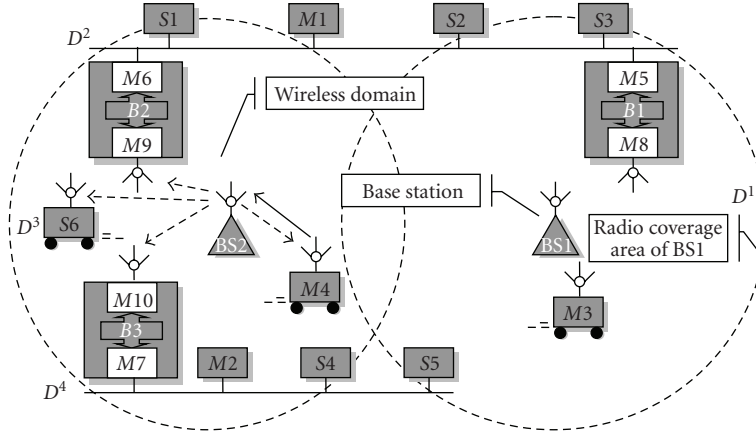


FIGURE 3: Bridge-based network example.

format and transmit those frames to the wireless domains and vice versa. Actually, the frame formats and the bit rates of the wired and wireless domains are different. The result of these characteristics is that queuing delays may appear at the repeaters. A solution to the problem relies on the manipulation of the PROFIBUS DLL  $T_{ID}$  parameters [17], by inserting an additional idle time before a master starts the transmission of a request frame, which guarantees that the repeater output queues do not increase in an undesirable way, compromising the real-time performance of the system. A consequence of this is that the setting of the  $T_{SL}$  parameter must be made in accordance with the new values for the  $T_{ID}$  parameter.

The mobility mechanism is based on the role of a specific master station—the Mobility Master (MM)—which periodically triggers the mobility management procedure by broadcasting a special frame—the Beacon Trigger (BT). The reception of this frame causes the BSs to start transmitting Beacon frames in their radio channels and wireless mobile stations start assessing the quality of all radio channels using the functionalities provided by their PhL and change to the best radio channel detected.

The network scenario depicted in Figure 2 comprises four domains: two wired domains ( $D^2$  and  $D^4$ ) and two wireless domains ( $D^1$  and  $D^3$ ). Three Repeaters ( $R_1$ ,  $R_2$ , and  $R_3$ ) interconnect the domains. The wireless communications are relayed by two BSs ( $BS_1$  and  $BS_2$ ). The network also comprises three wired masters ( $M_1$ ,  $M_2$  and  $M_5$ ), two wireless mobile masters ( $M_3$  and  $M_4$ ), five wired slaves ( $S_1$ ,  $S_2$ ,  $S_3$ ,  $S_4$ , and  $S_5$ ) and one mobile wireless slave ( $S_6$ ).

Finally, it is important to note that one of the main characteristics of the repeater-based approach is that it creates a “broadcast” network where the token rotates between all masters in the network (as depicted in Figure 2), and all messages are received by all stations in the network.

**3.3. Bridge-Based Architecture.** The first proposals for the use of a multiple logical ring PROFIBUS network have been described in [18, 19], the first proposed a multiple

logical ring topology for wired networks, and the second had the objective of providing differentiated QoS features to some stations. Later [20] had proposed to extend PROFIBUS to support wireless communications by the use of bridges.

In the bridge-based solution, bridges operate at Data Link Layer (DLL) level. Therefore, 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, a bridge needs to have two network interfaces, both supporting the same DLL. This means that such a dual-port PROFIBUS bridge would contain two master stations. Each master station that belongs to a bridge is referred as Bridge Master (BM). This station is a modified PROFIBUS master.

Figure 3 presents a bridge-based hybrid network example.

The network depicted in Figure 3 comprises three wired masters ( $M_1$ ,  $M_2$ , and  $M_5$ ), two wireless mobile masters ( $M_3$  and  $M_4$ ), five wired slaves ( $S_1$ ,  $S_2$ ,  $S_3$ ,  $S_4$ , and  $S_5$ ), and one mobile wireless slave ( $S_6$ ). Masters  $M_3$  and  $M_4$  are capable of moving between domains  $D^1$  and  $D^3$ . Three bridge devices are also considered ( $B_1$  ( $M_8 : M_5$ ),  $B_2$  ( $M_6 : M_9$ ),  $B_3$  ( $M_{10} : M_7$ )). Each bridge is composed by two BMs.

Network operation is based on the Multiple Logical Ring (MLR) approach as proposed in [7]. Therefore, each wired/wireless domain has its own logical ring. In this example, four different logical rings exist ( $D^1$  ( $M_3 \rightarrow M_8$ ),  $D^2$  ( $M_1 \rightarrow M_5 \rightarrow M_6$ ),  $D^3$  ( $M_4 \rightarrow M_9 \rightarrow M_{10}$ ) and  $D^4$  ( $M_2 \rightarrow M_7$ )).

Since one of the objectives of this work is to provide compatibility between standard wired PROFIBUS devices and the new wireless nodes we proposed two protocols, one for supporting the communication between stations in different domains—the Interdomain Protocol (IDP)—and another to support the mobility of wireless stations between different wireless domains—the Interdomain Mobility Procedure (IDMP).

**3.3.1. Interdomain Protocol (IDP).** The IDP explores some PROFIBUS-DP protocol features at the DLL and Application Layer (AL) level, which enables a master to repeat the same request until receiving a response from the responder station without generating errors to the upper layers.

When a master starts a transaction (Figure 4) with a station belonging to another domain, an Inter-Domain Transaction (IDT), it starts by transmitting a request frame addressed to the responder station (hereafter referred as IDreq frame). This frame is then relayed by only one of the BMs (denoted as  $BM_{ini}$ —ini stands for initiator) belonging to the initiator domain, according to its Routing Table (RT) information.  $BM_{ini}$  receives the IDreq frame, codes it according to the IDP, and stores internally information about the transaction, in a structure called List of Open Transactions (LOT). Additionally, a timer, the  $BM\_IDT\_Abort\_Timer$  ( $T_{BM-IDTAbort}$ ), is started. To distinguish from standard PROFIBUS frames we call frames coded according to the IDP as Inter-Domain Frames (IDFs).

The initiator periodically sends a request frame until receiving a response frame. Note that the AL of PROFIBUS-DP can operate as described without generating errors.

The IDreq frame is relayed by the bridges until reaching the last BM, which belongs to the responder domain (denoted as  $BM_{res}$ —res stands for responder).

IDFs are transmitted using the PROFIBUS Send Data with Acknowledge (SDA) service; consequently, when an error corrupts the frame, it can be retransmitted by the BM, increasing the reliability of the protocol. Nevertheless, this change requires that the BMs must receive the frame, decode its content, consult its RT, and send a confirmation, within the time allowed for the transmission of a confirmation frame ( $T_{SDR}$ ).

$BM_{res}$  decodes the original request frame and transmits it to the responder, which can be a standard PROFIBUS station (e.g., 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 address (SA) 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.

After that, the  $BM_{ini}$  is ready to respond to a repeated request from the initiator. The response frame is equal to the frame transmitted by the IDT responder. It is important to note that this somewhat inefficient behaviour is required to guarantee compatibility with standard PROFIBUS wired nodes.

Meanwhile, if the  $T_{BM-IDTAbort}$  expires, the related entry at the LOT is deleted and a new IDT can be reinitialised by the next initiator's request.

Figure 4 presents a simplified timeline of an IDT between master  $M3$  and Slave  $S6$ , in which we are assuming the network scenario presented in Figure 3.

In this example, a transmission error occurs when the IDF embedding the response is transmitted between  $BM M6$  and  $BM M5$ . Since the frame has not been acknowledged by

$BM M5$ ,  $BM M6$  retransmits the frame after the expiration of  $T_{SL}$ .

Note that the transaction between stations belonging to the same domain, Intra-Domain Transactions (IADTs), are handled by the standard PROFIBUS protocol.

### 3.3.2. Inter-Domain Mobility Procedure (IDMP)

**IDMP Overview.** The IDMP is a hierarchically managed procedure, where one master in the overall network—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). Finally, the BMs must implement specific mobility services.

The GMM must know the addresses of all the BMs and DMMs in the system and each DMM must know the addresses of the BMs that belong to its domain as well as those of the wireless mobile stations.

It is important to note that wireless mobile stations used in this approach only require the replacement of the existing wired PROFIBUS Physical Layer protocol, as proposed during the RFieldbus project [21]. These changes were based on the Direct Sequence Spread Spectrum (DSSS) technology used on the IEEE 802.11 Physical Layer, with additional functionalities to provide channel assessment for mobility proposes.

For example, and concerning the scenario illustrated in Figure 3,  $M6$  can assume both the role of GMM and DMM of domain  $D^2$ .  $BM M5$ ,  $M9$ , and  $M7$  assume the role of DMMs for wireless domain  $D^1$ , wireless domain  $D^3$ , and wired domain  $D^4$ , respectively.

The IDMP evolves through 4 phases ensuring that (i) it does not generate errors, (ii) that the inaccessibility periods are minimal (especially for IADTs), and (iii) that the wireless mobile stations are able to evaluate all wireless radio channels and switch to the best one (Figure 5).

This mechanism is synchronous in Phase 1 and Phase 2 as well as at the beginning of Phase 3. But the end of Phase 3 is not synchronized. Additionally, Phase 4 runs asynchronously for each domain.

Obviously, any wireless communication system is more prone to errors than a wired one; therefore the IDMP is provided with a set of error handling functionalities.

**IDMP Error Handling Overview.** The IDMP error handling mechanism is based on timers, which control the IDMP phases.

Four timers are assigned to the GMM, one to each BM and one to each DMM present in the network. Two of the timers associated to the GMM are used to detect and handle errors during Phase 1, while the other two are related to Phase 2. The timers associated with Phase 1 are designated as  $GMM\_Phase\_1\_Alert\_Timer$  ( $T_{GMM-P1Alert}$ ) and  $GMM\_Phase\_1\_Abort\_Timer$  ( $T_{GMM-P1Abort}$ ). The timers associated to Phase 2 are designated as  $GMM\_Phase\_2\_Alert\_Timer$  ( $T_{GMM-P2Alert}$ ) and  $GMM\_Phase\_2\_Abort\_Timer$

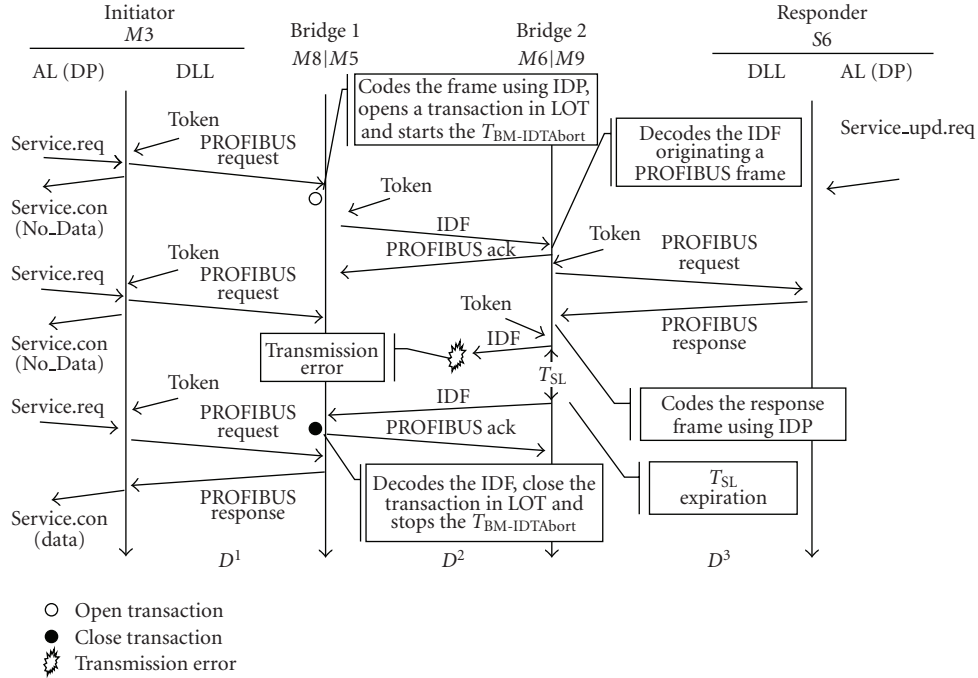


FIGURE 4: IDT between M3 and S6.

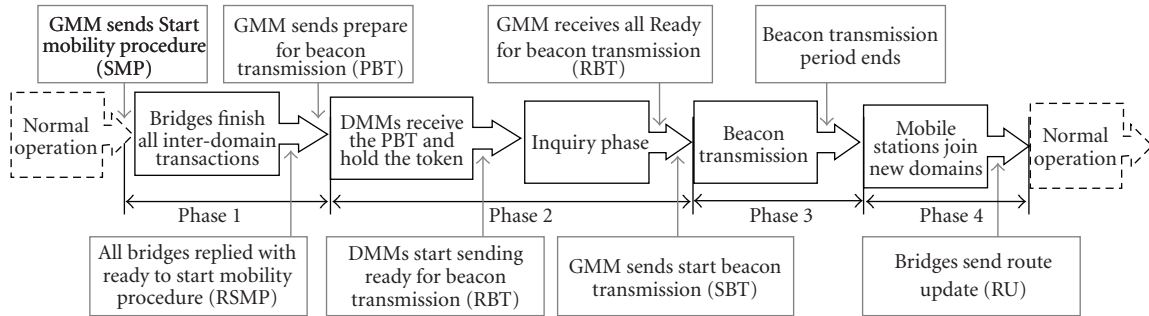


FIGURE 5: Inter-Domain Mobility Procedure Phases.

( $T_{GMM-P2Abort}$ ). The timers associated to BMs and DMMs are designated as  $BM\_IDMP\_Abort\_Timer$  ( $T_{BM-IDMPAbort}$ ) and  $DMM\_IDMP\_Abort\_Timer$  ( $T_{DMM-IDMPAbort}$ ), respectively.

The role of the management agents (GMM, DMMs, and BMs), the purpose of each timer, and the different phases of the proposed handoff mechanism are described in the following section.

### 3.4. Details on the IDMP

**Phase 1.** Periodically Phase 1 starts with a *Start\_Mobility\_Procedure* (SMP) message sent by the GMM; see Figure 6. At this point, the  $T_{GMM-P1Alert}$  and  $T_{GMM-P1Abort}$  are started. When the BMs receive an SMP 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 IDFs originated in other domains.

After completing all pending IDTs, the BMs transmit a *Ready\_to\_Start\_Mobility\_Procedure* (RSMP) message to the GMM.

If the GMM receives RSMP messages from all BMs in the network before the expiration of the  $T_{GMM-P1Alert}$ , it stops both timers. If  $T_{GMM-P1Alert}$  expires, that is, if it did not receive a RSMP message from at least one of the BMs in its list, it retransmits the SMP message and waits for the reception of a RSMP message from the BMs in lack. If it receives all RSMP from the BMs before the expiration of the  $T_{GMM-P1Abort}$ , it evolves to Phase 2. Otherwise, the IDMP is aborted and will be restarted on the next mobility period.

Additionally, each BM starts its  $T_{BM-IDMPAbort}$  when it receives the first SMP message. If a BM has already sent a RSMP message and it receives again an SMP message, it replies again with an RSMP message. If Phase 2 does not start before the expiration of  $T_{BM-IDMPAbort}$ , the BM aborts the IDMP and starts accepting new IDTs.

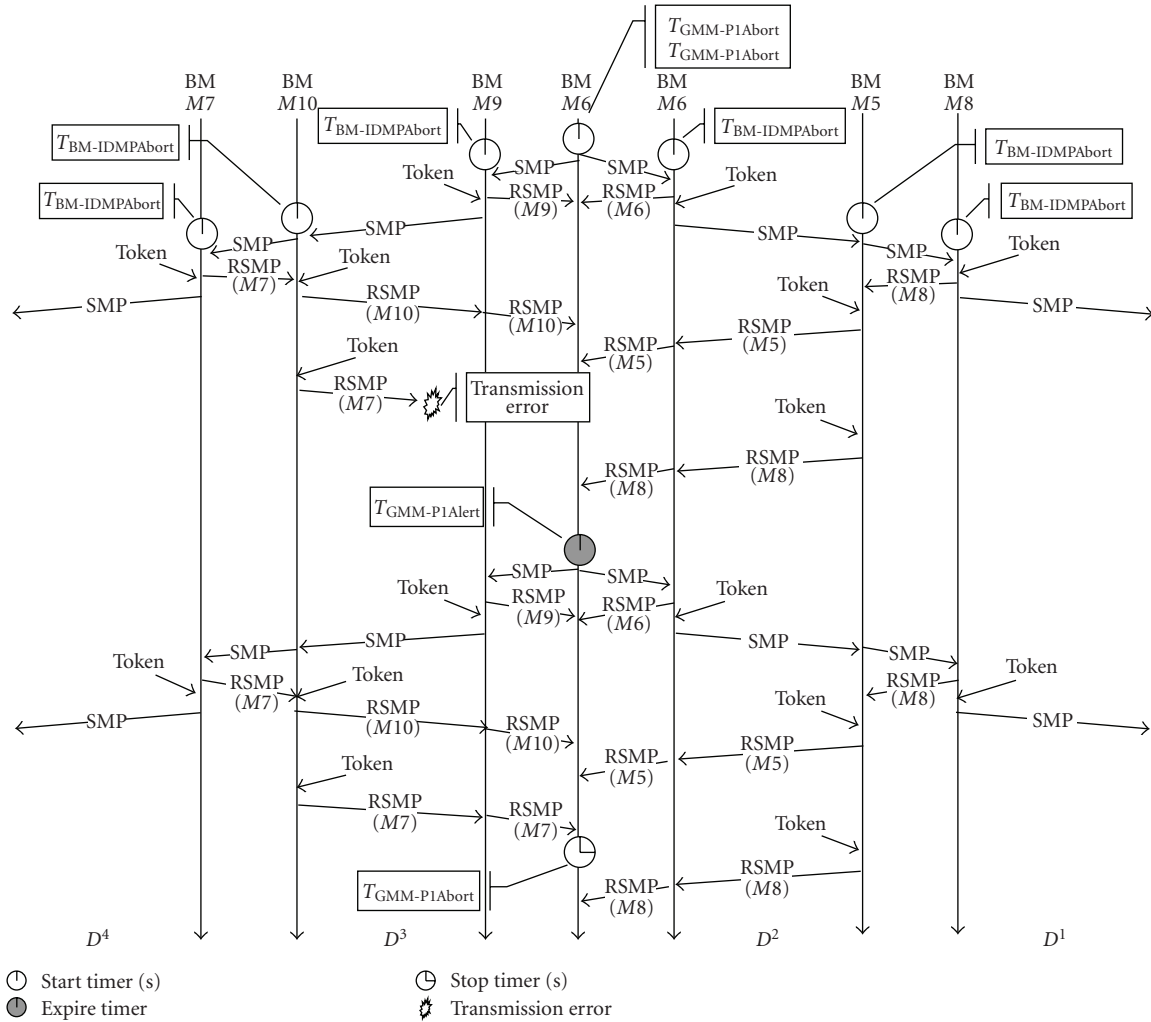


FIGURE 6: Simplified timeline for Phase 1.

*Phase 2.* Phase 2 is triggered by the GMM broadcasting the `Prepare_for_Beacon_Transmission` (PBT) message and it starts the  $T_{GMM-P2Alert}$  and  $T_{GMM-P2Abort}$  timers. After receiving the PBT message, a DMM starts the  $T_{DMM-IDMPAbort}$  and retains the token (after token reception, obviously), starting the inquiry subphase. At this point a `Ready_for_Beacon_Transmission` (RBT) message is transmitted to the GMM signalling that the DMM is ready for Beacon transmission. The BMs clear their RT entries related to wireless mobile stations at the reception of the PBT message.

The GMM stops the  $T_{GMM-P2Alert}$  and  $T_{GMM-P2Abort}$  if it receives an RBT message from all DMMs in the network, after the IDMP can evolve to Phase 3. Otherwise, the GMM broadcasts again a PBT message at expiration of the  $T_{GMM-P2Alert}$ . If the  $T_{GMM-P2Abort}$  expires before the reception of all RBT messages, then the IDMP is aborted.

On the inquiry subphase, 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 already on their output queues.

This procedure allows a faster communication between the GMM and the DMMs; at the same time, the inaccessibility period of wired stations is kept small. Phase 2 ends when all RBT messages are received by the GMM.

*Phase 3.* Phase 3 starts when the GMM transmits the `Start_Beacon_Transmission` (SBT) message; see Figure 7. Upon reception of this message, the DMMs and the BMs stop the  $T_{DMM-IDMPAbort}$  and  $T_{BM-IDMPAbort}$  timers, respectively, and the DMMs start emitting Beacon frames. The wireless mobile stations use the Beacon frames to evaluate the quality of the different radio channels and to decide if they want to handoff (or not). So, before the end of the Beacon transmission, every wireless mobile station that wants to handoff must switch to the new radio channel.

If a DMM on a wireless domain does not transmit Beacon messages, then the wireless mobile stations present in its wireless domain are not able to perform radio channel assessment and consequently stay in the same domain until the next IDMP. If a DMM does not receive an SBT, then the IDMP ends when the  $T_{DMM-IDMPAbort}$  expires and it sends a

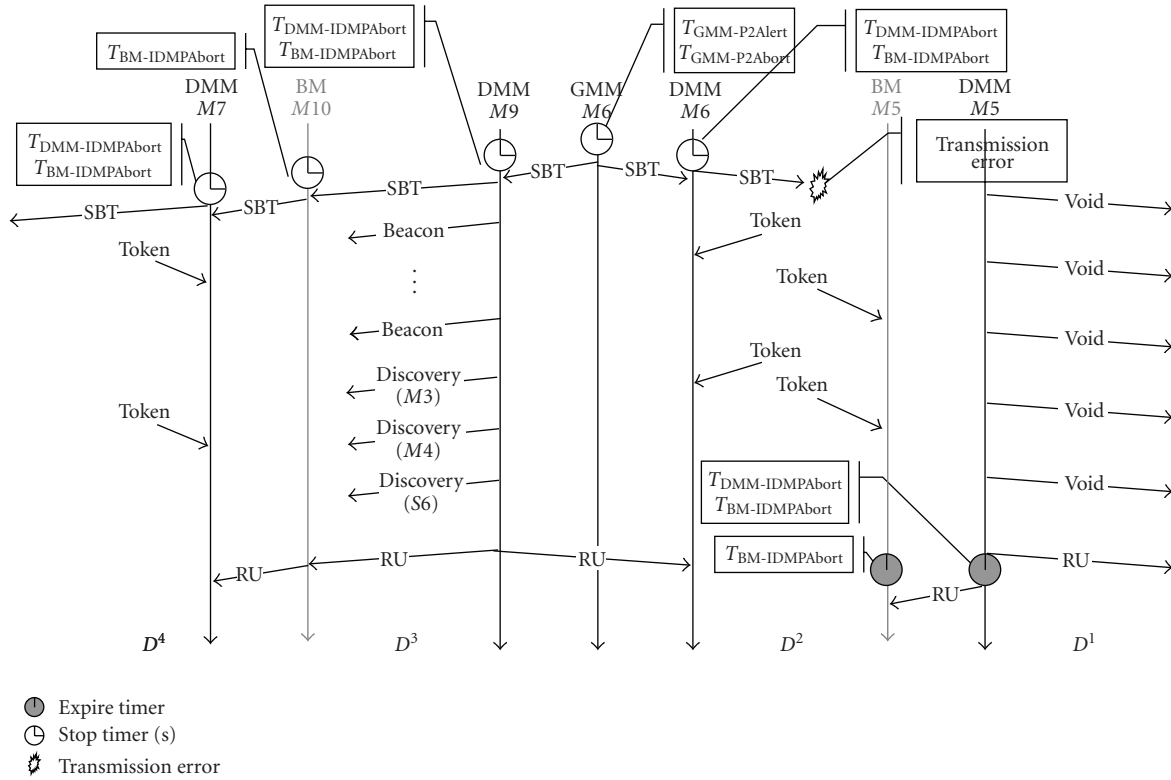


FIGURE 7: Simplified timeline for Phase 3 and Phase 4.

Route.Update (RU) messages related to the wireless mobile stations present in its domain. In this way, the BMs, which have cleared all entries related to the wireless mobile stations from their RTs at the reception of the PBT message, may update its RT.

*Phase 4.* After the end of the Beacon transmission, every wireless DMM (still holding the token) inquires all wireless mobile stations in order to detect if they are present in its domain, using Discovery messages. After finishing, the DMM broadcasts RU messages containing information about the discovered wireless mobile stations, permitting the BMs to restart routing IDTs related to wireless mobile stations.

Figure 7 presents a simplified timeline of Phase 3 and Phase 4.

In Section 6 the behaviour of these enhancements is analysed and compared with the repeater-based approach.

**3.5. Implementation Issues.** The main advantage of these protocols is that they enable a simple solution to maintain compatibility with standard PROFIBUS stations, putting the development effort only on the development of the bridges and on the development of Physical Layer extension to support radio communication and mobility on mobile wireless stations.

Figure 8 illustrates the main building blocks of a two-port bridge. In order to support the required functions,

there must be a set of mechanisms related to the IDP and to the IDMP. These mechanisms operate at DLL level and consequently the existing PROFIBUS DLL must be adapted. The operations of the IDP and IDMP are managed by three components: BM, DMM, and GMM.

The BM component, which gives to the device its name and is mandatory, contains the routing and the IDP handling functions which are crucial to the IDP and to the IDMP. These two functions are associated with five data structures: the Routing Table (RT), the List of Open Transaction on  $BM_{ini}$  (LOT INI), the List of Open Transaction on  $BM_{res}$  (LOT RES), the List of Active Stations in Domain (LASD), and the List of Wireless Mobile Stations in the Network (LWMSN).

Frames are relayed by a BM according to the information contained in its RT. The LOT INI is used to store information about ongoing IDTs in which the BM assumes the role of  $BM_{ini}$  and the LOT RES is used when a BM assumes the role of  $BM_{res}$ . The LASD is a list of all masters and slaves that belong to the BM domain and is used by the BM to know if it is the transaction's  $BM_{ini}$  and  $BM_{res}$  or simply a bridge in the path between the initiator and the responder. The LWMSN is a list of addresses of all wireless mobile stations present in the network and is used in the IDMP.

The other two components, DMM and GMM, are optional and their functions are related to the IDMP. The DMM functionalities require two data structures: the List of Bridge Masters in the Domain (LBMD) and the



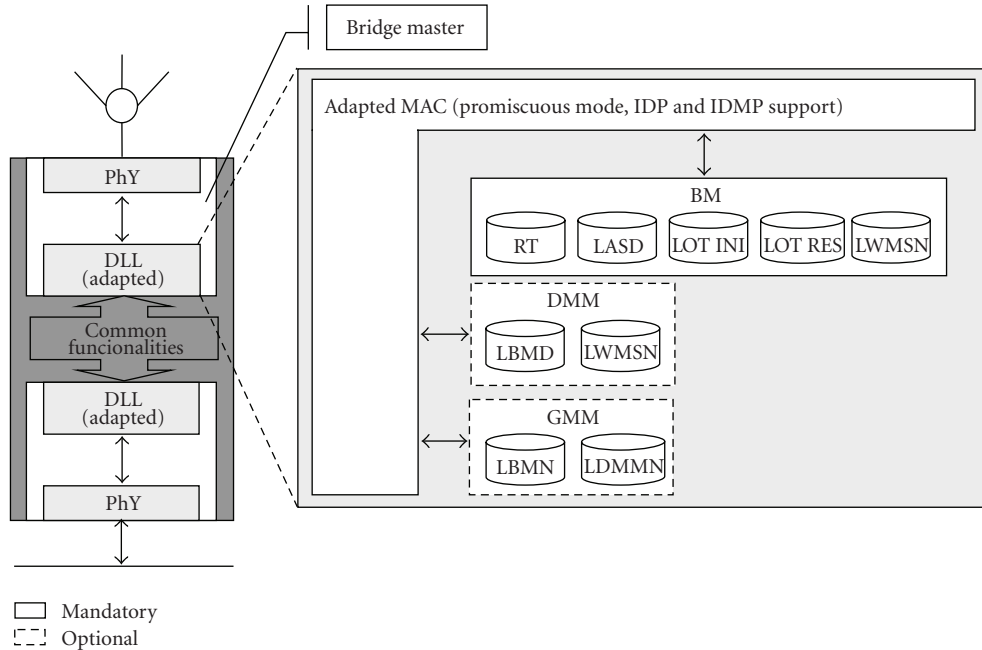


FIGURE 8: Bridge architecture.

LWMSN. The LBMD is a list that contains the domain’s BMs addresses and is used in the IDMP inquiry subphase. The LWMSN is a list of addresses of all wireless mobile stations present in the network and is used in the IDMP discovery subphase.

The GMM must also be provided with two data structures: the List of Bridge Masters in the Network (LBMN) and the List of Domain Mobility Managers in the Network (LDMMN). The LBMN is a list of address of all BMs present in the network and is used to control the received *Ready\_to\_Start\_Mobility\_Procedure* (RSMP) messages during the IDMP Phase 1. The LDMMN contains all network DMM addresses and is used to control the received *Ready\_for\_Beacon\_Transmission* (RBT) messages during the IDMP Phase 2.

Figure 8 also shows the Common Functionalities (ComFunc) box, which is supported by a shared memory area and is responsible for the communication between the two BMs of a bridge; however, if necessary this functionality can support more than two ports. For further details on this implementation the reader is referred to [22].

#### 4. Simulator Implementation

All mechanisms proposed for the bridge-based approach were validated using a simulation model. In this section, we detail the implementation of the simulator that implements the bridge-based approach, called Bridge-Based Hybrid Wired/Wireless PROFIBUS Network Simulator (BHW2PNetSim). We had developed a simulation model for the repeater-based, the Repeater-Based Hybrid Wired/Wireless PROFIBUS Network Simulator (RHW2PNetSim) [23]. Note that both simulators share

some modules, since they were developed using the same tools. Supported by these two implementations we were able to compare the behaviour of both approaches.

Our simulation models can be classified as dynamic, discrete, and stochastic, according to the definitions presented in [24]. We had chosen to use the Objective Modular Network Testbed in C++ (OMNeT++) [25] framework to implement the simulation model. The main reason for our choice is that this tool offers some of the necessary functionalities to implement such complex systems, from scratch.

OMNeT++ is an object-oriented modular discrete event. The basic system components (*simple* modules) are built using the C++ programming language and then assembled into larger components (*compound* modules) and complex models (*system* module) using a high-level language, named NED (an OMNeT++ specific scripting language). An OMNeT++ model consists of hierarchically nested modules that communicate among them using messages which represent frames or packets in a computer network. These messages can contain arbitrarily complex data structures. Simple modules send messages through *gates* or directly based on their unique identifier. Messages can arrive from another module or from the same module (self-messages are used to implement timers).

Gates are the input and output interfaces of modules. Messages are sent out through output gates and arrive through input gates. Each connection (also called a link) is created within a single level of the module hierarchy and is composed by two gates.

The simulation executable is a standalone program, which can be run in any machine. When the program is started, it reads from a configuration file (usually *omnetpp.ini*) the model parameters for each simulation run.

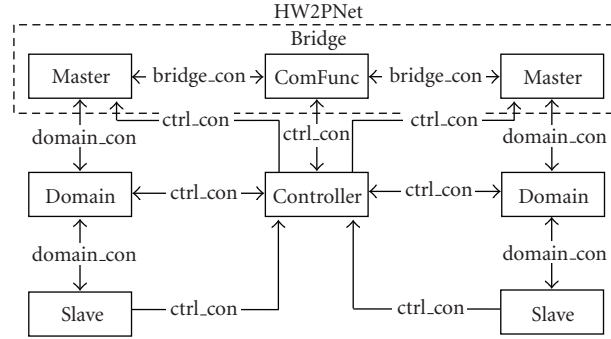


FIGURE 9: Modules and connections of the BHW2PNetSim.

```

...
theBHW2PNet.controller_domain="\
<d><n>D1</n><m>M8:M3</m><s></s><dmm>M8</dmm><pos>400:300</pos></d>\
<d><n>D2</n><m>M6:M1:M5</m><s>S1:S2:S3</s><dmm>M6</dmm><pos>200:150</pos></d>\
<d><n>D3</n><m>M9:M10:M4</m><s>S6</s><dmm>M9</dmm><pos>50:300</pos></d>\
<d><n>D4</n><m>M7:M2</m><s>S4:S5</s><dmm>M7</dmm><pos>250:450</pos></d>"

theBHW2PNet.controller_inter_domain="\
<b><n>B1</n><m>M8:M5</m><pos>350:150</pos></b>\
<b><n>B2</n><m>M6:M9</m><pos>50:150</pos></b>\
<b><n>B3</n><m>M10:M7</m><pos>120:400</pos></b>"
...

```

FIGURE 10: Configuration file related to the Controller module instance (excerpt).

**4.1. Bridge-Based Simulator Architecture.** The Bridge-Based Hybrid Wired/Wireless PROFIBUS Network Simulator (BHW2PNetSim) implements the simulation model of the bridge-based approach. The BHW2PNetSim is composed by 6 main OMNeT++ modules: HW2PNet, Controller, Domain, ComFunc, Master, and Slave.

Figure 9 shows how the main modules are interconnected. There are 3 kinds of the connections: `ctrl_con`, `domain_con`, and `bridge_con` connections. The `ctrl_con` connections are used to establish the connections between the Controller module instance and all module instances in the overall system. This kind of connection has no delay and is used for control and configuration purposes. The `domain_con` connections are used to establish the connections among all domain components (between Master and Slave module instances and the Domain module instance). The `bridge_con` connection supports the bridge's connections. In practice a bridge is an abstraction, which is composed by two Master module instances connected by a ComFunc module instance.

In OMNeT++ to actually get a simulation that can be run, it is necessary to write a network definition. A network definition declares a simulation model as an instance of the system module, in this case of the HW2PNet module.

The Controller is a simple module that coordinates the simulation and performs several managing tasks, acting as the simulation supervisor. Parameters that are specific of one module instance or common to all module instances in the network are assigned to the Controller module instance and, on simulation setup, the Controller module

instance makes the parameter setting of the all other module instances. Additionally, due to memory limitations, the Controller module instance is responsible for periodically sending commands to other module instances, commanding them to dump the information gathered to data files.

It is important to note that the simulator log files can have a size of hundreds of MB.

Finally, whenever a Master or Slave module instance changes between domains, this module updates the network configuration and the corresponding connections. Note that OMNeT++ does not provide any native mechanism for mobility.

In order to illustrate the Controller module configuration tasks, Figure 10 presents an excerpt of its configuration. It represents the setting of two parameters called `_domain` and `_inter_domain`. These parameters are strings, which define the configuration of the domains and the bridges. Both of them are written using a predefined structure based on tags. The parameter values present in Figure 10 configure the network depicted in Figure 3.

The meaning of the tags used in `_domain` parameter are the following: `<d>` and `</d>` specify a domain; the tags `<n>` and `</n>` enclose the name of the Domain module instance; `<m>` and `</m>` enclose the name of the masters belonging to the domain, which are separated by a colon; `<s>` and `</s>` tags are similar to the previous case but associated with slaves; `<dmm>` and `</dmm>` define the Master module instance that is the DMM of the domain; `<pos>` and `</pos>` indicate the position of the domain for graphical representation purposes. Note that coordinate

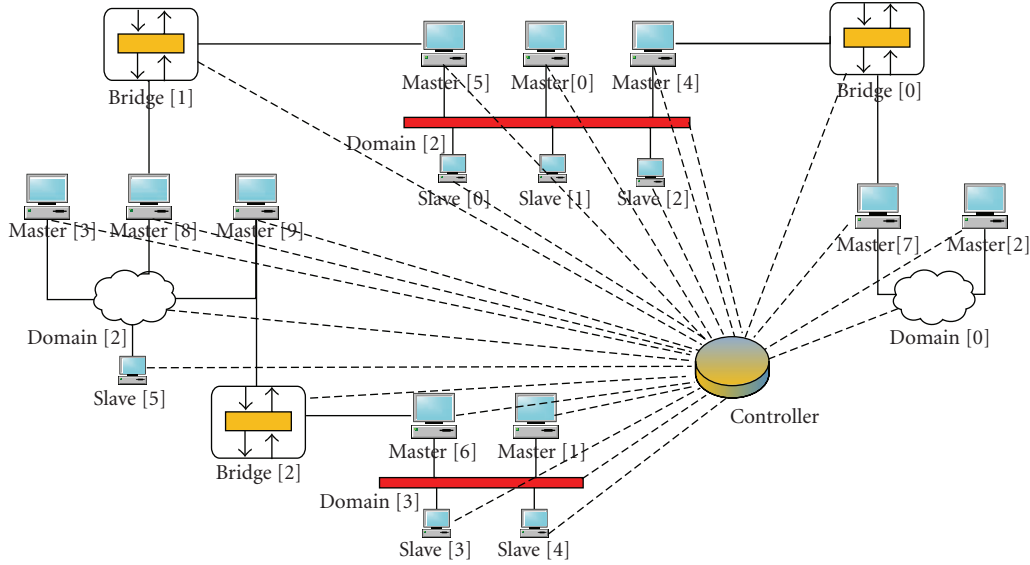


FIGURE 11: Screenshot of the output window of the BHW2PNetSim.

(0, 0) represents the top-left corner of the window as shown in Figure 10. In this particular case, the first domain (" $\langle b \rangle \langle n \rangle B1 \langle /n \rangle \langle m \rangle M5 : M8 \langle /m \rangle \langle pos \rangle 350 : 150 \langle /pos \rangle \langle /b \rangle$ ") is referred to as  $D1$  and is composed by two Master module instances (which are named  $M3$  and  $M8$ ); no Slave module instances are defined in this domain. Its DMM is the Master module instance named  $M8$ . The Domain module instance is depicted in the screen at position (400, 300).

The parameter `_inter.domain` defines the configuration of a bridge. The meaning of the tags IS the following:  $\langle b \rangle$  and  $\langle /b \rangle$  define a bridge;  $\langle n \rangle$  and  $\langle /n \rangle$  are used to set the name of the ComFunc module instance; between tags  $\langle m \rangle$  and  $\langle /m \rangle$  enclose the names of the Master module instances composing a bridge separated by colon;  $\langle pos \rangle$  and  $\langle /pos \rangle$  indicate the position of the ComFunc module instance for graphical representation purposes.

The first bridge presented in Figure 10 (" $\langle b \rangle \langle n \rangle B1 \langle /n \rangle \langle m \rangle M5 : M8 \langle /m \rangle \langle pos \rangle 350 : 150 \langle /pos \rangle \langle /b \rangle$ ") is referred to as  $B1$  and it is composed by two Master module instances ( $M5$  and  $M8$ ), which are depicted at position (350,150). This bridge interconnects two domains  $D1$  and  $D2$ .

The Controller module instance uses the `_domain` and `_inter.domain` parameters information to perform the parameterization of the module instances, such as the LAS and GAPL of each Master, the RT of each BM, the LBMD of each DMM, and the LDMMN of a GMM, just to mention some parameters. It also stores, in internal variables, the structure of the network. Using this information the network configuration can be changed when a Master or Slave module instance moves between wireless domains.

Figure 11 shows a graphical representation used by the simulator to represent the network scenario shown in Figure 3.

In, Figure 11 it is clear that the Controller module instance (labelled *controller*) is able to communicate with all module instances, for parameterization and simulation control purposes. Master (labelled *master*[ $x$ ], where  $x$  is a number between 0 and 10) and Slave (labelled *slave*[ $x$ ], where  $x$  is a number between 0 and 5) module instances are connected to their correspondent Domain module instances, symbolized by a rectangle (labelled *domain*[1] and *domain*[3]) or a cloud (labelled *domain*[0] and *domain*[2]) for the case of wired or wireless domains, respectively. This network scenario is also composed by three bridges. The ComFunc module instance of each bridge is labelled *bridge*[ $x$ ], where  $x$  can be 0, 1, or 2.

**4.1.1. Broadcast Support.** In spite of the OMNeT++ capacities, only one-to-one connections are supported. One-to-many and many-to-one connections can only be achieved using special purpose simple modules. Therefore, it was necessary to develop a simple module—the Domain module which is able to connect all stations in a domain and simulate a broadcast network. The connections are created and assigned dynamically enabling the support of mobility. This is done through the `domain_con` connections.

The ComFunc module instance establishes connections between the Master module instances that belong to the same bridge through the `bridge_con` connections (Figure 9).

**4.1.2. Station Modules.** The PROFIBUS master and slave stations are modelled by the Master and Slave modules.

A Master module is a compound module that maps a master station. It is composed by three modules: Master\_PHY, Master\_DLL, and Msg\_Stream. In each Master module instance there is one instance of Master\_PHY and Master\_DLL modules. The number of the Msg\_Stream module instances can be from 1 up to 64 (Figure 12).

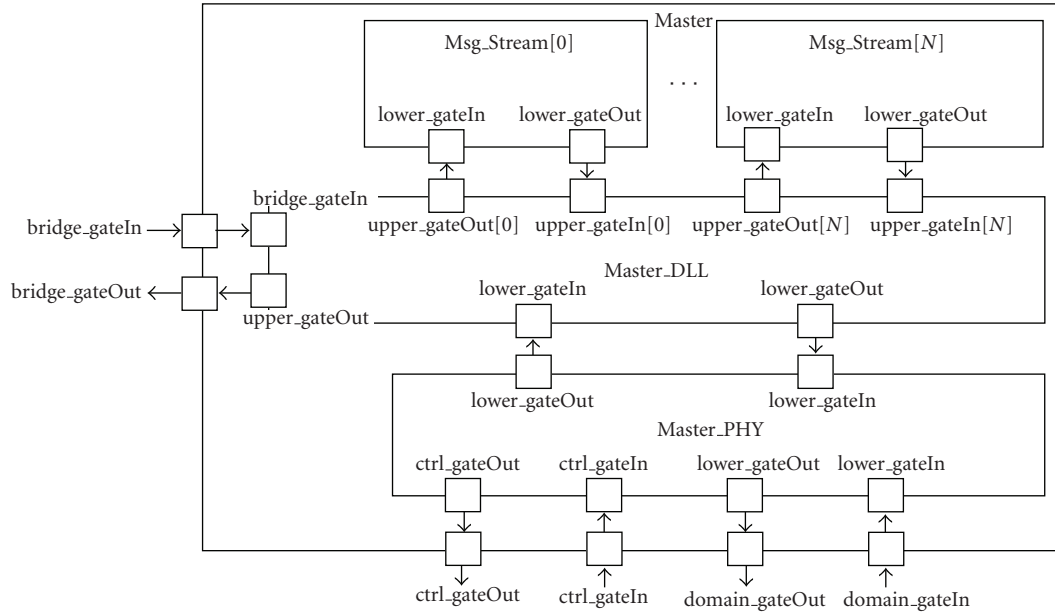


FIGURE 12: OMNeT++ Master module composition.

The `Master_PHY` module models the PhL of the PROFIBUS protocol. It represents the network interface of the Master module; it receives messages from a Domain or from a Controller module instance and passes the messages to the `Master_DLL` module and vice versa.

The `Master_DLL` module models the PROFIBUS DLL and the necessary functions to support part of the IDP and IDMP functionalities. Consequently, a Master module, besides modelling a simple PROFIBUS DLL master, can also operate as a BM and/or as a DMM and/or as a GMM. For that reason, the `Master_DLL` module is a compound module composed by four simple modules: DLL, BM, DMM, and GMM, as shown in Figure 13.

The DLL module models the PROFIBUS DLL as well as the required adaptations in order to support the IDP and the IDMP. BM, DMM and GMM modules model IDP and IDMP agents.

The BM module is a simple module that models the mechanisms necessary for the IDP and the IDMP-related functions. The DMM module is a simple module that models the DMM functions required by the IDMP. The GMM module is also a simple module that models the GMM required functionalities.

The `Msg_Stream` module models the typical behaviour of the AL. It can be configured to periodically request services from the `Master_DLL` module instance, which can easily be configured on the PROFIBUS communication stack.

A `Slave` is a compound module which maps into a standard PROFIBUS slave station. It is structured similarly to the `Master` module.

**4.1.3. Mobility Modelling.** In order to simulate the mobility of the wireless mobile stations, `Master` and `Slave` modules have a parameter called `_location_vector`.

The `_location_vector` is a string that defines the location of each `Master` and `Slave` module instance during time. In order to limit the size of the configuration files used, the `_location_vector` parameter is written in a compact format. Each location is represented by a tuple  $(n_{mob}, Dx)$ , where  $n_{mob}$  represents the number of mobility procedures during which the `Master` or `Slave` module instance stays on domain  $Dx$ .

Figure 14 depicts part of the configuration file related to a `Master` module instance, which models a wireless mobile station called `M3`. This station stays in domain `D1` for five mobility procedures, and then it changes to domain `D3` where it will stay for another 10 mobility procedures. This sequence of events repeats itself until the end of the simulation.

The information to set the `_location_vector` parameter can be obtained using the Mobility Simulator [26]. This simulator models the radio wave propagation according to the Log-normal Shadowing model [27] and the mobility of wireless mobile stations.

## 5. Analytical Real-Time Analysis

Automation systems are very prone to timing errors, since some of them must operate at very high rates. At such rates any timing error might result on the nonexecution of an operation over a product or on the damaging of the product. Assume as an example a glass handling machine, which requires to handle glass without breaking it. Timing guarantees depend not only on the software running generally on special purpose Programmable Logical Controllers (PLC) but also on communication delays between sensors, actuator, and the PLCs controlling the system. With this objective in mind the designers of such systems must be

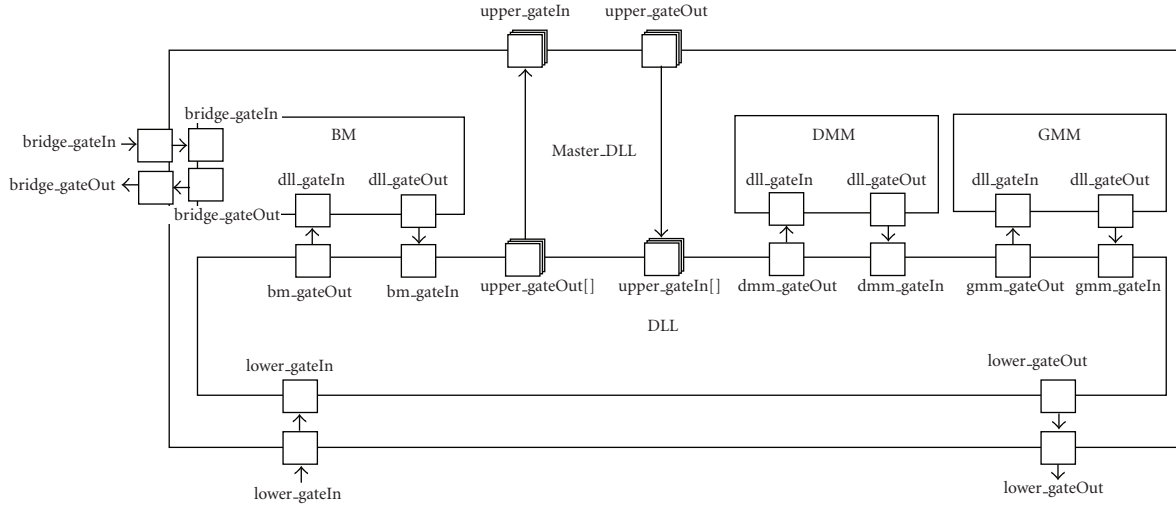


FIGURE 13: OMNeT++ Master\_DLL module composition.

```

...
theBHW2PNet.master[2]..name="M3"
theBHW2PNet.master[2]..vector_location="5,D1:10,D3:"
...
    
```

FIGURE 14: Configuration file related to the Master module instance (excerpt).

capable of guaranteeing, during the system design phase, that the timing constrains (usually referred as deadlines) are assured by the system.

This objective can be assured by special purpose mathematical tools which describe the computation and network delays in such system. In the remainder of this section we focus on the network delays, related to the bridge-based approach.

**5.1. Message Model.** We define a Message Stream as a periodic sequence of message transactions, related with specific system functionality, for example, 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 when a request is transmitted in unicast or broadcast mode, the initiator does not expect any response; that is, the case of the Send Data with No acknowledge (SDN) service.

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, and  $P_i^k$  represents the priority of the message stream. PROFIBUS only defines two priority levels: high and low.

The worst-case duration of a complete message transaction (SRD 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

$$C_i^k = \text{treq}_i^k + \max T_{\text{SDR}} + \text{tres}_i^k + T_{\text{ID1}}, \quad (1)$$

where  $\text{treq}_i^k$  and  $\text{tres}_i^k$  represent the time required to transmit the request and response frames, respectively. These two latencies can be calculated using appropriate equations.

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:

$$C_i^k = \text{treq}_i^k + T_{\text{ID2}}. \quad (2)$$

**5.2. IDT and IADT Worst-Case Response Time.** Related to the timing analysis approach presented in [28], the WCRT for a high-priority message stream  $i$  from a master  $k$ , in an SLR network ( $\text{Rslr}_i^k$ ), or in the case of the bridge-based approach referred as an Intra-Domain Transaction (IADT), can be computed by

$$\text{Rslr}_i^k = nh^k \times T_{\text{cycle}}^k + Ch_i^k, \quad (3)$$

where  $nh^k$  is the number of synchronous high-priority message streams generated in master  $k$  and  $Ch_i^k$  is the worst-case duration of a synchronous message cycle  $i$  issued by master  $k$ .  $T_{\text{cycle}}^k$  is the worst-case token rotation time, which can be computed according to the analysis proposed in [28].

In [29], we proposed a worst-case timing analysis of the IDP. Relevant to that analysis is the fact that the initiator of the IDT needs to periodically repeat the request until getting the actual response from the  $\text{BM}_i$  (Figure 4). Consequently, the WCRT for a message stream  $i$  from master  $k$  on an MLR

network ( $Rmlr_i^k$ ) can be formulated as follows:

$$Rmlr_i^k = A_i^k \times T_i^k + Rslr_i^k, \quad (4)$$

where  $A_i^k$  is the maximum number of attempts required to obtain the actual response, which depends on the delay experienced by the IDT, from the reception of the request at the  $BM_i$ , until the arrival of the respective response to the  $BM_i$  ( $Rbmi_i^k$ ). Therefore,  $A_i^k$  can be obtained by computing  $[(Rslr_i^k + Rbmi_i^k - C_i^k)/T_i^k]$ .  $Rbmi_i^k$  can be obtained as follows:

$$Rbmi_i^k = \sum_{f=2}^{b+1} (Rslr_i^k)^{\Omega_{req}[f]} + \sum_{f=2}^b (Rslr_i^k)^{\Omega_{res}[f]} + 2 \times b \times \phi, \quad (5)$$

where  $b$  is the number of bridges between the initiator and the responder.  $\Omega_{req}$  represents the set of BMs which relay the IDT request frame in the path from the initiator to the responder.  $\Omega_{res}$  represents the set of BMs which relay the IDT response frame in the path from the initiator to the responder. The network domains are numbered from 1 to  $b+1$ .  $\Phi$  represents the relaying delay imposed by the bridges.

The mobility-related messages are transmitted using the PROFIBUS DLL SDN service, which only involves the transmission of a request message; consequently (5) must be updated, taking into account that the worst-case time required by a request from a message stream  $i$ , to go from a master  $k$  to another station  $w$  ( $Ru_i^{k-w}$ ), is given by

$$Ru_i^{k-w} = Rslr_i^{k'} + \sum_{f=1}^b Rslr_i^{\Omega_{req}[f]} + (b + di + df) \times \phi, \quad (6)$$

where  $k'$  is the first BM to transmit the request, which can be master  $k$  itself, when it is directly connected to the first domain in the path (in this case  $di$  is equal to 0), or can be the BM on the other side of the bridge if master  $k$  is a BM not directly connected to the first domain in the path (in this case  $di$  is equal to 1).  $df$  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.

**5.3. Taking into Account Mobility.** 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 met. 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 subphase and the Beacon transmission subphase, IADTs are disabled.

The timing analysis presented for IDTs in Section 5 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.

A comprehensive WCRT analysis has been presented in [30] but due to its complexity in this paper we do not elaborate further.

Basically, the delays defer on the combination of involved nodes, which can be wired, fixed wireless, or mobile wireless nodes. It also depends on the kind of transaction and even on message stream period. To the purpose of this paper we will denote the mobility-related delay by  $D_{mob}$  and update (2) as follows:

$$Rmlr_{m_i^k} = A_i^k \times T_i^k + Rslr_i^k + D_{mob}. \quad (7)$$

It is important to note that (3) to (7), represent the worst-case scenarios and they can be pessimist. As an example, (3) represents a worst-case scenario since it considers that a station can simultaneously have in its output queue the maximum number of message streams; this equation is also pessimist since the calculation of  $T_{cycle}^k$  assumes that all message streams have the same message transmission time. Consequently, the cases which are assumed by these equations can be extremely rare or might even never happen. But they provide a base over which to build very reliable applications.

## 6. Network Scenario

The use of simulation tools can be an easy and cost effective method to guarantee the schedulability of an industrial network. In relation to an analytical analysis it has the advantage of guaranteeing the setting of the application based on statistical results.

In this section we describe the network scenarios used to do a comparative analysis, between the repeater and the bridge-base approach. We also use the same scenario to show how the analytical analysis compares with simulation results and how it is used to perform the setting of some BM parameters on the bridge-based approach.

**6.1. Network Base Configuration Scenario.** The comparison study is based on the network scenarios presented in Figures 2 and 3.

It is assumed that wireless domains,  $D^1$  and  $D^3$ , use the 802.11b Direct Sequence Spread Spectrum (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 wired domains,  $D^2$  and  $D^4$ , use a standard PROFIBUS PhL operating at 1.5 Mbit/s and 0.5 Mbit/s; for domains  $D^2$  and  $D^4$ , respectively, each character is coded using 11 bits according to the RS485 standard. In wired domains, the PhL frames do not have a head or a tail sequence.

To perform the simulations we have assume that some parameters can be stochastically modelled using a triangular distribution function [31]. Therefore, in the remainder of the test we are using the following notation: *triang(minimum, apex, maximum)*, where *minimum* and *maximum* represents the minimum and maximum values of the distribution and *apex* the median for the distribution.

TABLE 1: Repeater-based scenario parameters (bit times).

Domain	$T_{ID1}$	$T_{ID2}$	$T_{SL}$	$T_{TR}$
$D^1$ and $D^3$	2132	1088	2856	39712
$D^2$	1447	740	2142	27520
$D^4$	100	100	714	4858

6.1.1. *Repeater-Based Scenario.* The station addresses have been set according to the following:  $(M1, 1)$ ,  $(M2, 2)$ ,  $(M3, 3)$ ,  $(M4, 4)$ ,  $(M5, 5)$ ,  $(S1, 41)$ ,  $(S2, 42)$ ,  $(S3, 43)$ ,  $(S4, 44)$ ,  $(S5, 45)$ , and  $(S6, 46)$ . Consequently, the Highest Station Address (HSA) master parameter has been set equal to 5 in all masters.

The  $T_{ID}$  and  $T_{SL}$  and the parameters related to the Beacon message were calculated with the help of the RFieldbus System Planning application, described in [30].

In this approach, the  $T_{ID2}$  parameter on M5 (2677 bit times) must be made different from the other stations, since it has the role of Mobility Manager.

In order to guarantee that, at the token arrival, there is always enough time to execute all pending high-priority traffic, the master  $T_{TR}$  parameter has been set according to the formulations proposed in [28], assuming an error-free medium. Table 1 resumes the settings of the  $T_{ID1}$ ,  $T_{ID2}$ ,  $T_{SL}$ , and  $T_{TR}$  master DLL parameters in each domain.

6.1.2. *Bridge-Based Scenario.* We have assumed that the  $T_{SDR}$  parameter can be modelled stochastically using a triangular distribution function with apex at 70 bit times and extremes at 11 and 100 bit times.

The internal delay of the ISs ( $\Phi$ ) is equal to 30 microsecond and the `max_retry_limit` parameter has been set to 1. The mobility procedure is triggered every 200 millisecond. Another important detail concerns the Gap Update factor (G), which is set to 1 in all domains, in order to have the GAP Update mechanism always active and reduce the delays associated with the mobility procedure.

The stations timing parameters can be set according to the recommendation of the PROFIBUS standard [1]; therefore the  $T_{ID}$  and the  $T_{SL}$  parameters have been set to 100 and 115 bit times, respectively; the  $T_{TR}$  parameter has been set to 2076 bit times in wireless domains and 2046 and 1306 bit times in wired domains  $D^1$  and  $D^4$ , respectively.

As mentioned, to handle and detect errors in the IDP and IDMP the bridge-based approach is provided by a set of timers. The setting of the error handling mechanism timers can be done supported by the worst-case timing analysis, resumed in Section 5. Nevertheless, such a setting, based on pessimistic scenarios, would negatively affect the network behaviour in normal operating conditions. Consequently, we based our setting on a mixture of simulations results considering an error-free environment and in our worst-case timing analysis.

To detect and handle errors during the execution of an IDT, we have set the  $T_{BM-IDTAbort}$  equal to 33.022 ms, which is the WCRT of a message transaction considering that the IDMP is inactive.

TABLE 2: Message streams.

Msg. St.	Parameters	Msg. St.	Parameters
$S_1^{M1}$	(S1, 15, 20, high, 100)	$S_1^{M3}$	(S4, 15, 20, high, 100)
$S_2^{M1}$	(S2, 15, 20, high, 100)	$S_2^{M3}$	(S6, 15, 20, high, 100)
$S_3^{M1}$	(S5, 15, 20, high, 100)	$S_3^{M3}$	(S3, 15, 20, high, 100)
$S_1^{M2}$	(S3, 15, 20, high, 100)	$S_1^{M4}$	(S6, 15, 20, high, 100)
$S_2^{M2}$	(S6, 15, 20, high, 100)		

According to our simulation results, the maximum duration of IDMP Phase 1 and IDMP Phase 2 is 11.156 ms and 4.090 ms, respectively. Since these results were obtained considering an error-free medium, these values were used to set the alert and the abort timers,  $T_{GMM-P1Alert}$  (11.156 ms) and  $T_{GMM-P1Abort}$  (22.312 ms), respectively. The abort timer was set to the double of the alert timer, since this setting permits the completion of Phase 1, even when the SMP message is not received by any BM. The same rules were used to set the  $T_{GMM-P2Alert}$  and the  $T_{GMM-P2Abort}$  to 4.090 ms and 8.181 ms respectively.

The  $T_{BM-IDMPAbort}$  was set equal to 30.490 ms, which is the sum of  $T_{GMM-P1Alert}$  and  $T_{GMM-P2Alert}$ . The  $T_{DMM-IDMPAbort}$  was set equal to  $T_{GMM-P2Alert}$  (8.181 ms) because this timer is started at the beginning of Phase 2 and finishes at the end of Phase 2.

The mechanisms used by PROFIBUS to detect and handle error situations can degrade the availability of the bridge-based network. To decrease the impact of the PROFIBUS token recovery mechanism we defined a set of rules for master address assignment.

- (1) The lowest address in each logical ring must be given to a BM;
- (2) The following addresses should be separated by two or more units between them.

The first rule guarantees that the token recovery mechanism is always performed by a BM. The second rule is used to reduce the time that a station is out of the logical ring, for further details and justifications see [32]. Therefore, on a bridge-based network the master's address has been set to:  $(M1, 5)$ ,  $(M2, 3)$ ,  $(M3, 9)$ ,  $(M4, 10)$ ,  $(M5, 4)$ ,  $(M6, 2)$ ,  $(M7, 1)$ ,  $(M8, 7)$ ,  $(M9, 6)$ , and  $(M10, 8)$ . This solution requires the setting of a different HSA in every domain.

6.2. *Message Streams.* The set of message streams presented in Table 2 tries to illustrate some probable transaction scenarios in the network. The message streams are specified as tuples (destination address, request frame length (in bytes), response frame length (in bytes), and priority and deadline (in ms)). The deadline for all message streams is equal to 100 ms.

During simulation, to emulate, as close as possible, the condition usually found in real-scenarios, the period and offset of each stream had been set according to the following. For the master to whom we want to perform the measurements, the message stream periods were set to a constant value. Contrarily, for the other masters, the message

stream parameters were set using a triangular distribution function. The period of the message streams being measured has been set to 8 ms with no initial offset and the period and the initial offset. The other message streams have been set using  $triang(7.8, 8, 8.2)$  and  $triang(0, 7.8, 8)$ , respectively.

The results for each master message stream set have been obtained as the aggregate result of 100 runs, each with 120-second duration. These simulations have been performed in an 8-core machine (with two Intel Xeon Processor L5310 processors) and it required more than 24 hours to complete all simulation runs, generating more than 2 GB of data.

## 7. Results Discussion

In this section, we present and analyse some simulation results upon variation of the network parameters: bit rate and maximum frame size. We also compare them with worst-case results.

**7.1. Worst-Case Response Time.** The calculation of the WCRT for the system message streams is a complex process which takes into account the number and type of stations in each domain, the domains involved on the relaying of messages related to each message stream and the delays due to the mobility procedure. Table 3 presents these results. In this table each message stream is represented by  $S_x^{My}(D^{d1}, D^{d2})$ , where  $D^{d1}$  and  $D^{d2}$  represent the domain in which the transaction initiator and responder are located.

An obvious conclusion, which can be withdrawn by comparing the results with  $(Rbmi_i^k)$  and without mobility  $(Rmlr_m_i^k)$ , is that the proposed mobility procedure introduces significant delays on the message streams, in some cases there is an increase by more than four times, but this is a consequence of the fact that our proposal builds on top of the existing PROFIBUS protocol and only requires changes on the physical layer and some additions on the DLL of the bridges.

Streams  $S_1^{M1}$  and  $S_2^{M1}$  are related to IADTs. Streams  $S_3^{M1}$  and  $S_1^{M2}$  are related to IDTs involving resident wireless stations or wired stations (station which do not move between different wireless domains). The other streams involve mobile wireless stations. Message stream  $S_2^{M2}$  involves a wired master and a mobile wireless slave; therefore it is strongly influenced by the IDMP; it is also interesting to note that the response varies according to the domain in which the mobile station is located. Streams  $S_1^{M3}$  and  $S_3^{M3}$  involve a mobile wireless master and a wired slave.

All WCRT analyses assume pessimistic and extreme conditions which can lead to response times many time higher than average and maximum response times measured in practice or by simulation.

To compare the simulation results with the analytical results, three message streams are used on this study, one IADT ( $S_1^{M1}$ ), one IDT ( $S_1^{M2}$ ), and one IDT involving two mobile wireless stations ( $S_2^{M3}$ ).

A histogram is shown on Figure 15 comparing the response time calculated using the worst case formulations (labelled as WCRT) with the maximum response time

TABLE 3: 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	6.53
$S_2^{M1}$	—	3.33	6.53
$S_3^{M1}$	10.70	20.33	65.39
$S_1^{M2}$	13.26	28.90	79.94
$S_2^{M2}(, D^1)$	19.07	31.90	100.49
$S_2^{M2}(, D^3)$	5.51	17.90	76.49
$S_1^{M3}(D^1)$	23.86	35.75	91.87
$S_1^{M3}(D^3)$	1.17	9.55	61.20
$S_2^{M3}(D^1, D^3)$	16.03	24.75	80.56
$S_2^{M3}(D^3, D^1)$	11.14	19.55	77.93
$S_3^{M3}(D^1)$	6.24	15.75	66.08
$S_3^{M3}(D^3)$	6.24	15.55	63.00
$S_1^{M4}(D^1, D^3)$	16.03	33.56	91.23
$S_1^{M4}(D^3, D^1)$	11.14	29.42	86.96
$S_1^{M4}(D^1)$	6.24	15.75	66.08
$S_1^{M4}(D^3)$	6.24	15.55	63.00

obtained from simulation (labelled as SR). As it would be expected the worst-case response time computed according to the formulations presented in Section 5 is much higher than the simulation results, particularly when considering IDTs. It is important to note that the simulation results are a summary of millions of transaction and hundreds of simulations runs, in many different conditions.

**7.2. Error-Free Environment.** In this section, we present and analyse some simulation results upon variation of some network parameters: bit rate, ISs internal delay, and maximum frame size.

The message streams used on the comparison between these two approaches were  $S_1^{M1}$ ,  $S_1^{M2}$ , and  $S_3^{M3}$ , one IADT and two IDTs, respectively, where  $S_3^{M3}$  involves mobile stations (master  $M3$  and slave  $S6$ ).

In the repeater-based scenario there was the need to adjust the period of the message streams, because with some parameter setting, the network enters into saturation since the network is used beyond its maximum throughput.

**7.2.1. Base Configuration Results.** This subsection discusses the results obtained using the base configuration described in Section 6.1.

Figure 16 shows a histogram of the measured response time values for  $S_1^{M1}$  in both scenarios. Note that, in the subtitle of this figure and on the remaining figures in this Section, an  $R$  or a  $B$  before the message stream symbol ( $RS_i^k$  and  $BS_i^k$ ) specify that the values are related to the repeater-based or to the bridge-based architecture, respectively.

In the repeater-based scenario, the minimum response time (MinRT) value is equal to 1.23 ms and the maximum response time (MaxRT) value to 16.09 ms.



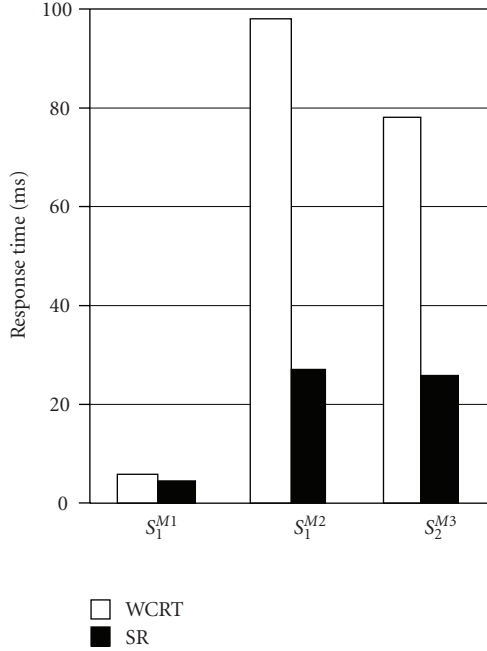


FIGURE 15: Comparison between simulation results (SRs) and WCRT results (WCRT).

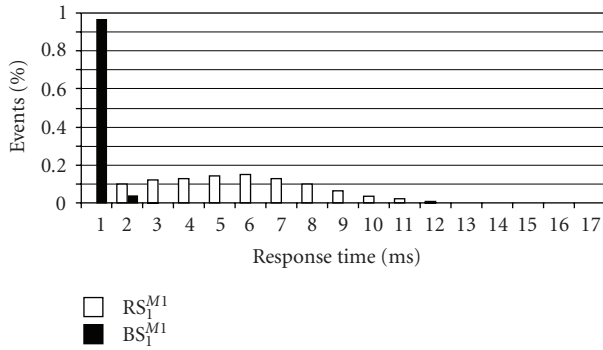


FIGURE 16: Response time histogram for the message stream  $S_1^{M1}$ .

In the bridge-based scenario, the MinRT value and the MaxRT value of message stream  $S_1^{M1}$  are 0.27 and 4.27 ms, respectively. Nevertheless, it is important to note that 96.20% of the transactions present a response time smaller or equal to 1 ms and 98.25% of the transactions have a response time value smaller than 1.23 ms, which is the MinRT of the repeater-based scenario. In this scenario,  $S_1^{M1}$  benefits from the smaller setting of the  $T_{ID}$  parameters as well as from the traffic segmentation resulting from the use of the bridges. The first reduces the message cycle duration, while the second reduces the traffic within domain  $D^1$ .

Figure 17 depicts a response time histogram for message stream  $S_1^{M2}$  in both scenarios. The repeater-based scenario presents the MinRT (1.22 ms) and MaxRT (18.49 ms) values smaller than in the bridge-based scenario. The MinRT and MaxRT values, in the bridge-based scenario, are 8.80 ms and 26.80 ms, respectively.

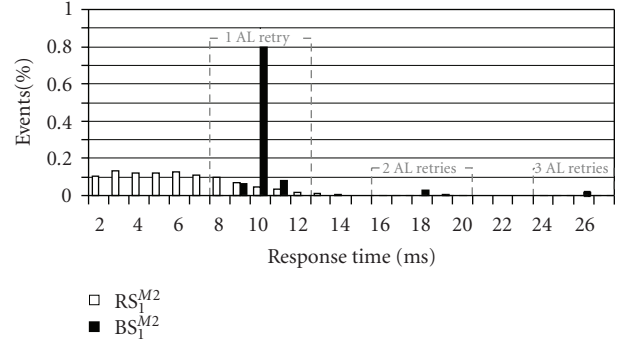


FIGURE 17: Response time of the message stream  $S_1^{M2}$ .

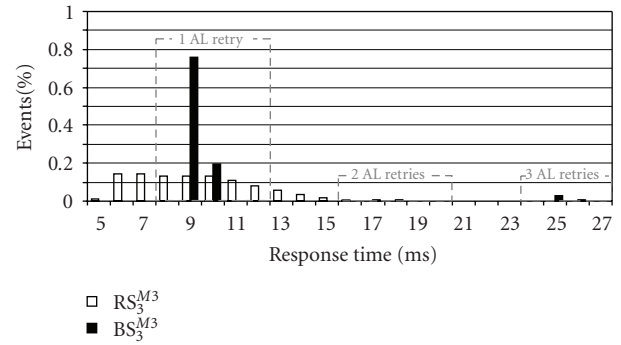


FIGURE 18: Response time histogram for the message stream  $S_3^{M3}$ .

In the repeater-based scenario, the histogram for  $S_1^{M2}$  is similar to the histogram of  $S_1^{M1}$  as it would be expected, since the use of repeaters creates a broadcast network.

In the bridge-based scenario, the timing behaviour of message stream  $S_1^{M2}$  is different than for  $S_1^{M1}$ , since  $S_1^{M2}$  is an IDT. Therefore, such kind of transaction requires that the initiator performs at least one Application Layer (AL) retry before obtaining a response (meanwhile stored at the  $BM_{ini}$  ( $BM_{M7}$ )). The period of this message stream is equal to 8 ms, and consequently the MinRT value in the bridge-based scenario is greater than 8 ms. It is noticeable that 94.23% (which is sum of the percentage in the intervals  $[8-9] \dots [11-12]$ ) of the transactions require only one AL retry and 4.13% (which is sum of the percentage in the intervals  $[16-17] \dots [19-20]$ ) of the transactions required two AL retries and the remaining (1.64%) three AL retries. The mean response time (MeanRT) value is equal to 10.02 ms on bridge-based scenario.

The response time histogram of message stream  $S_3^{M3}$  is shown in Figure 18. The results are very similar to message stream  $S_1^{M2}$ . However, in the repeater-based scenario the MinRT (4.61 ms) and MaxRT (19.85 ms) values are higher than the MinRT and MaxRT of message streams  $S_1^{M1}$  and  $S_1^{M2}$ .

The main reason for these results is due to the simulation model in which message stream  $S_3^{M3}$  is always queued in third place on  $M3$ 's output queue. Therefore, frames related to message stream  $S_3^{M3}$  have to wait for the transmission of frames related to the other two message streams in which the initiator is  $M3$ . This operation mode is similar to the

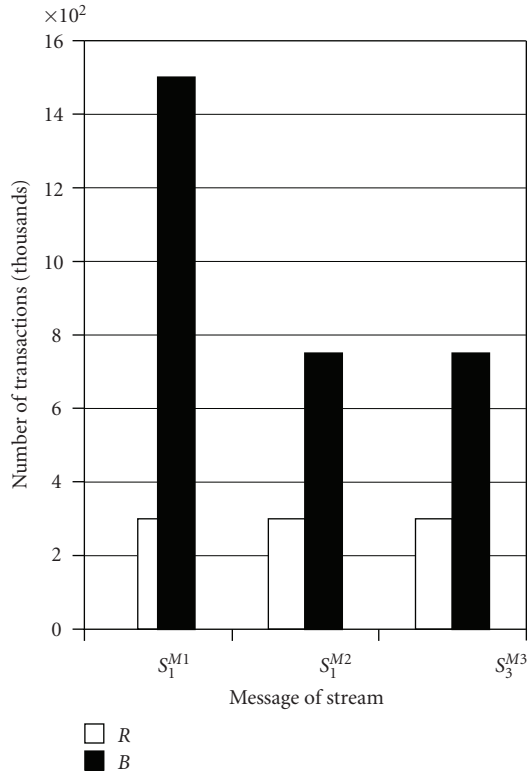


FIGURE 19: Number of message stream transactions.

typical behaviour of a Programmable Logical Controller (PLC) running PROFIBUS.

In the bridge-based scenario, the MinRT and MaxRT values for message stream  $S_3^{M3}$  are 8.66 ms and 26.08 ms, respectively. These results are similar to the results presented by message stream  $S_1^{M2}$  as would be expected since both are IDT.

It is important to note that if message stream  $S_3^{M3}$  had been queued in first place, instead of third, the results obtained, in the repeater-based scenario, would be equal to 1.41 ms and 16.30 ms, for MinRT and MaxRT, respectively. In the bridge-based scenario the results would be equal to 8.18 ms and 25.05 ms, for MinRT and MaxRT, respectively. From this results we conclude that the message streams queuing order has higher influence in the repeater-based scenario than in the bridge-based scenario, due to the fact that a single transaction in the repeater-based scenario takes much more time.

It is also important to note that the bridge-based scenario presents a much higher throughput than the repeater-based scenario. Figure 19 shows a histogram related to the number of transaction for each message stream.

In the bridge-based scenario the number of transactions for message stream  $S_1^{M1}$  is approximately 500% more; for message streams  $S_1^{M2}$  and  $S_3^{M3}$  the ratio drops to 240% more since these are IDTs. The main reasons for this disparity in results are (i) the traffic segmentation provided by the MLR approach, (ii) the lower overhead caused by the IDTs, (iii) the smaller settings of  $T_{ID}$  parameter, and; (iv) the message

stream period which is much lower (8 ms and 40 ms for the bridge and repeater-based scenarios, resp.).

In the following subsections, we will analyse the network timing behaviour when network parameters are varied.

**7.2.2. Variability of the Message Stream Response Time as a Function of the Bit Rate.** This subsection analyses how the setting of different bit rates in some network domains affects the timing behaviour of the two approaches. For this purpose, the results presented were obtained by varying the bit rate in domain  $D^4$ .

Figure 20 compares the MinRT, MeanRT and MaxRT values of the two scenarios for messages streams  $S_1^{M1}$  and  $S_3^{M3}$ , assuming the base configuration described in Section 6.1 by varying the bit rate in domain  $D^4$  from 0.5 Mbit/s to 5 Mbit/s.

In these conditions, parameters  $T_{SL}$ ,  $T_{ID1}$  and  $T_{ID2}$  must be recalculated for every bit rate in the repeater-based approach and these changes are applied to all domains. In the bridge-based scenario the parameter changes only affect domain  $D^4$ .

In the Figure 20 and in the following figures of this section the MinRT, MeanRT and MaxRT values are identified by a dash. The MinRT and MaxRT values are placed on the lower and upper extremes of the line and the MeanRT is placed between MinRT and MaxRT using a wider dash.

From the observation of Figure 20, we can conclude that in the repeater-based scenario the variability of the bit rate in domain  $D^4$  has a strong influence on response time of these message streams. In this scenario, the lower MaxRT occurs when  $D^4$  is operating at 1.5 Mbit/s but it keeps increasing afterwards. The main reason for this behaviour is due to the need of inserting an additional idle time to compensate the dissimilarities of the bit rates.

In the bridge-based scenario the bit rate variation in domain  $D^4$  has a small influence on the response time values of message streams  $S_1^{M1}$  and  $S_3^{M3}$ , since these message streams are not relayed by domain  $D^4$ . The decrease verified in the MaxRTs value when the bit rate increases is mainly due to a reduction of the IDMP-related latencies.

Again in this case, in the bridge-based scenario the number of concluded transaction is almost more 500% for IADTs, and 240% for IDTs than in the repeater-based scenario. For instance, the number of transactions for message streams  $S_1^{M1}$  and  $S_3^{M3}$  in the bridge-based scenario, considering a bit rate in domain  $D^4$  of 5 Mbit/s, is 1 500 000 and 722 900, respectively. In the repeater-based scenario, the number of transactions is 300 000 for both message streams.

**7.2.3. Variability of the Message Stream Response Time as a Function of the ISs Delays.** The ISs delay is the time required by an IS (bridge or repeater) to relay a frame between the domains to which it connects. In the repeater-based approach it is the time required by the repeater to convert between frame formats. In the bridge-based approach it is the time required to take routing decisions, for the conversion of frame formats and for its queuing on the output queue of the other BMs of a Bridge.

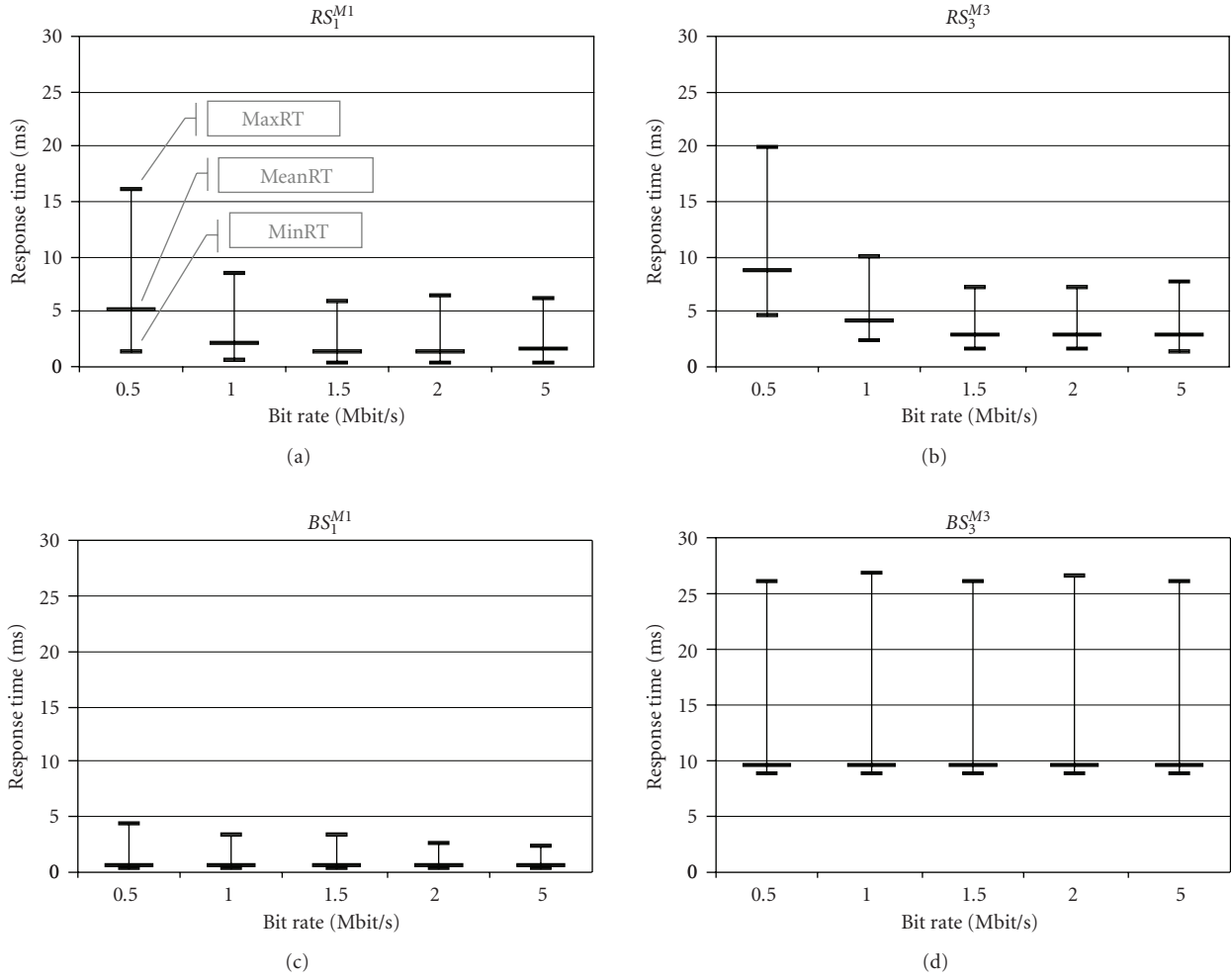


FIGURE 20: Influence of  $D^4$  bit rate on the message stream response time values.

In order to analyze the ISs internal delay influence on the network timing behaviour we performed six simulations in which the internal delay varied between 30 and 1000  $\mu$ s.

In the repeater-based scenario, there was the need to increase the message streams period to 80 ms since with higher values of the internal delay (500 and 1000  $\mu$ s) the network entered into saturation. The period for the other messages streams has been set using *triang*(78, 80, 82) and the offset has been set using *triang*(0, 78, 80).

Figure 21 depicts the MinRT, MeanRT, and MaxRT values for message streams  $S_1^{M1}$  and  $S_3^{M3}$  as a function of the ISs delays.

In the repeater-based scenario, the internal delay of the repeater has a stronger influence on the MeanRT and MaxRT values, due to the increase on the message cycle latencies. Additionally, the internal delay of the repeater requires a new setting of the  $T_{ID2}$  parameter of the MM, and consequently, the mobility procedure takes longer.

In the case of the bridge-based scenario, the internal delay of the bridge has a small influence on the response time values of message stream  $S_1^{M1}$  (an IADT), since the frames exchanged in these kind of transactions are not relayed by

bridges. The small MaxRT value increase is mainly due to the increase of the IDMP-related latencies. The effect on message stream  $S_3^{M3}$  (an IDT) is attenuated due to repetitions performed by the initiator until retrieving a response from the IDT  $BM_{ini}$ .

It is also important to note that the internal delay of the ISs has a strong influence in the repeater-based scenario throughput. As mentioned, there was the need to increase the message stream period to 80 ms, which is twice the message stream period of the base configuration. Consequently, the number of transaction decreased for half. For this reason, in the bridge-based scenario the number of concluded transaction is 1000% more for IADTs, and 480% more for IDTs.

**7.2.4. Variability of the Message Stream Response Time as a Function of the Maximum Frame Size.** The variation of the frame size impacts the duration of message transactions not only due to the increase on the message cycle time but also, in the case of the repeater-based approach, due to the need to changing some network timing parameters.

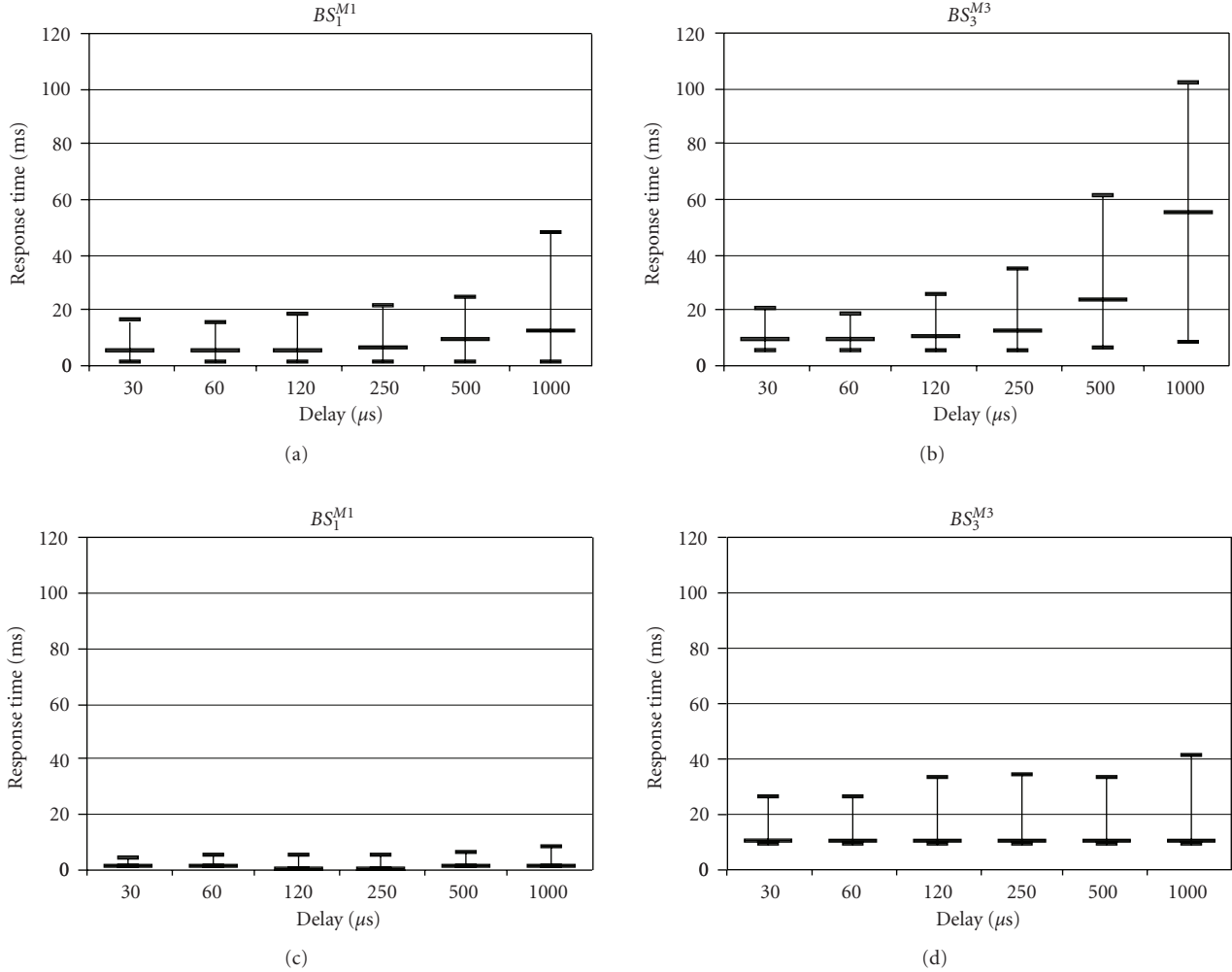


FIGURE 21: Influence of the IS delay on the response time.

To perform this comparison we have chosen to vary the frame size of message stream  $S_1^{M3}$ . This message stream is the first message stream of master  $M3$ , a wireless mobile station, and the responder is slave  $S4$ , which belongs to domain  $D^4$ . The size of the request and response frames varies between 15 and 250 bytes. Figure 22 depicts those results.

Once again, in the repeater-based scenario there was the need to increase the period to 160 ms and to adjust the  $T_{SL}$ ,  $T_{ID1}$ , and  $T_{ID2}$  parameters for every frame size. The period for the others messages streams was set using  $triang(140, 160, 180)$  and the offset was set using  $triang(0, 140, 160)$ .

All message streams are affected by the increase of the maximum frame size. In the bridge-based scenario, this influence is stronger for message streams which are routed through the same domains as  $S_1^{M3}$ , which is the case of message stream  $S_1^{M2}$ . But for  $S_1^{M1}$  that influence is very reduced; contrarily, in the repeater-based scenario all message streams are severely affected.

It was necessary to increase the message stream period in the repeater-based scenario to 160 ms; consequently, the

number of transactions performed in the bridge-based scenario is 2000% more for IADTs, and 943% more for IDTs. As an example, the number of transactions for message streams  $S_1^{M1}$  and  $S_1^{M2}$  in the bridge-based scenario considering a frame size of 250 Bytes is 1 500 000 and 707 809, respectively; in the repeater-based scenario the number of transactions is 75 000 for both message streams.

**7.2.5. Error Free Environment Conclusions.** From these experiments, we have noted that on the bridge-based approach the variability of the response time histograms is smaller than that in the repeater-based approach, although, in some cases, the maximum response time for IDTs can be superior.

The bridges permit a higher throughput of the overall network, which has been confirmed by our experiments, since in the bridge-based case the number of message transactions performed is in all cases much higher.

It is also noticeable that the messages queuing order has practically no influence in the maximum response time of a message stream in the bridge-based approach, contrarily to the repeater-based approach.

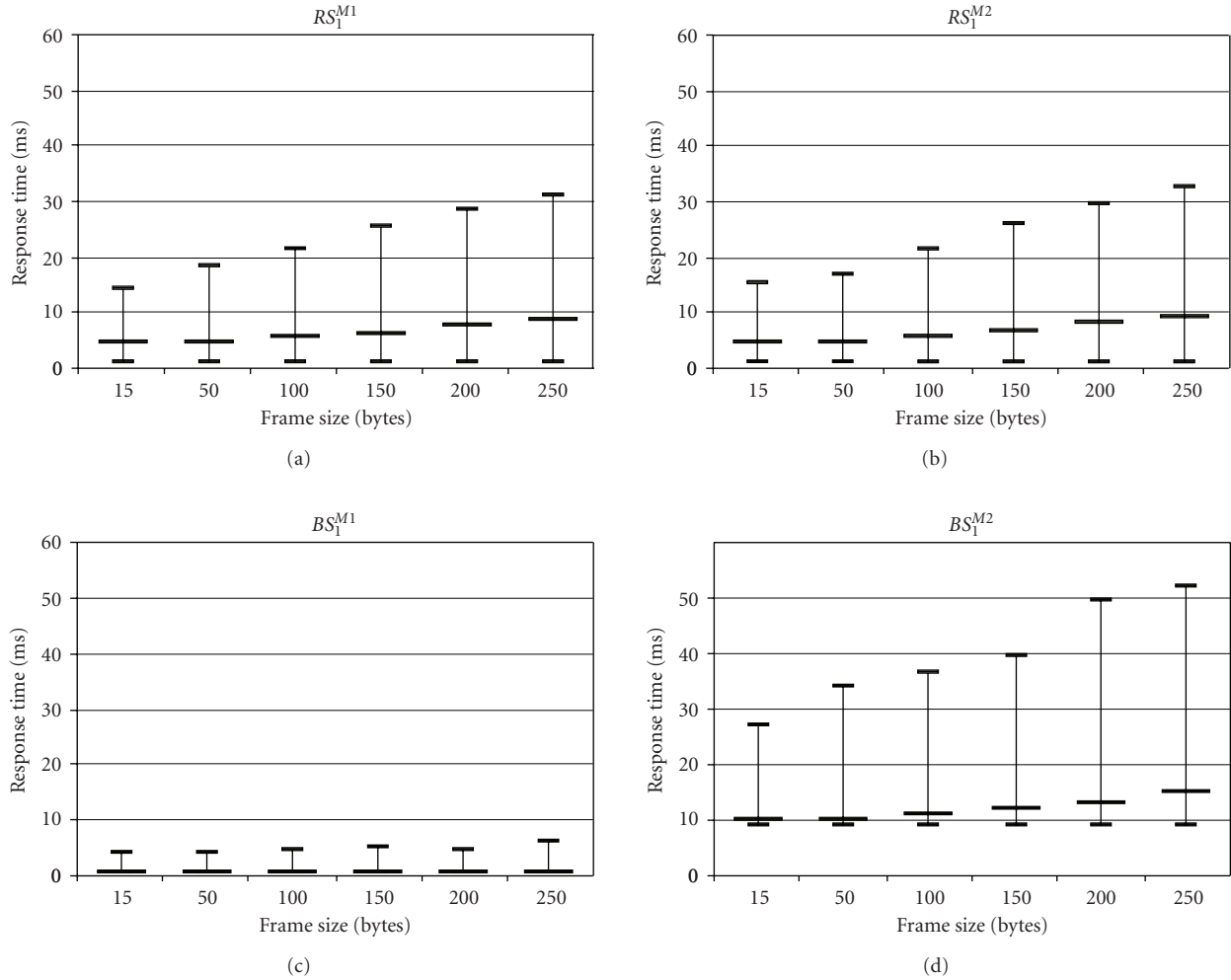


FIGURE 22: Influence of the maximum frame size on response time.

It can also be concluded that the repeater-based approach is more influenced by networks parameter changes, especially, when the maximum frame size in the network is increased.

Additionally, the mobility procedure used in the bridge-based approach leads to higher inaccessibility times for the wireless mobile stations, due to the complex mobility procedure required in order to maintain compatibility with existing hardware.

**7.3. Error-Prone Environment.** Wireless communications are usually more prone to errors than its wired counterparts, either due to sharing of the ISM band with other devices or due to electromagnetic interference. In this section we compare the simulation results considering an error-prone environment. The errors have been modelled using the well-known, Gilbert-Elliot error model [33, 34]. The main feature of this model is that it is able to simulate error burst conditions. For further details on this error model and on its parameter setting see [32].

We have chosen three sets of experiments. In the first we consider an error-free environment (hereafter this setting is

referred as scenario A). In the second, we consider that the mean Bit Error Rate (mBER) of the wired domains is equal to  $10^{-5}$  and the mBER of the wireless domains is  $10^{-6}$  (scenario B). In the last, we consider that the mBER of wired and wireless domains is  $10^{-4}$  and  $10^{-5}$  (scenario C), respectively.

This comparative analysis is based on four items. First, we analyse the number of token losses and the maximum time that a specific station requires to reenter into the logical ring. Second, the failures in the IDMP are analysed and compared. Third, we make a performance analysis based on the response time, the network throughput and the number of missed deadlines. Finally, we show the impact on the network performance of changing the frame length. In order to identify the simulation results, in the figures, we use an *R* for results from the repeater-based scenario and a *B* for results from the bridge-based scenario.

**7.3.1. Lost Token Events.** One cause of the network performance degradation is the token lost. Whenever a master is sending a token frame, it must hear from the medium all transmitted bits in order to detect a defective transceiver. If the token sender detects differences between the transmitted

TABLE 4: Token losses.

	A (%)	B (%)	C (%)
R	0	$1.32E-7$	$1.32E-5$
B	0	$1.68E-7$	$1.74E-5$

TABLE 5: Mobility procedure fails.

	A (%)	B (%)	C (%)
R	0	0,000117	0,14246
B	0	0,000100	0,79744

and the received frame in two consecutive transmissions, then it must remove itself from the logical ring. This situation is recovered by means of the timeout timer ( $T_{TO}$ ), already described in Section 2. If a period of inactivity longer than  $T_{TO}$  is detected, then the token is claimed by the master with the lowest address in the logical ring, and the logical ring is reinitialised. Table 4 shows the percentage of token losses in each scenario.

The percentage of the token losses in the bridge-based scenario is slightly higher than that in the repeater-based scenario. Three reasons explain that (i) there are more token frames transmitted and more masters; (ii) the token frame rotates faster (i.e., it is transmitted more frequently) in the bridge-based scenario than in the repeater-based scenario, due to  $T_{ID}$  setting; (iii) the error model used creates error bursts which have a higher probability of affecting two consecutive token transmissions when the timing parameters are smaller, as in the case of the bridge-based approach. However, the time required to detect and recover from a token loss event is much lower in the bridge-based scenario. As an example master M3 in average requires 97.988 ms to reenter into the logical ring in the repeater-based scenario, while in the bridge-based scenario it only requires 2.741 ms.

**7.3.2. Mobility Procedure Failures.** As outlined in Section 3.3.2 the bridge-based approach mobility procedure is much more complex, involving more message transfers, than the one of the repeater-based approach. This complexity causes more mobility procedure failures, as it is confirmed by the results presented in the Table 5, especially for scenario C.

**7.3.3. Network Performance.** Another objective of our study was to determine how the network performance would be affected by errors. Figure 23 depicts a graphic of three message streams response times on both scenarios: one IADT ( $S_1^{M1}$ ), one IDT ( $S_1^{M2}$ ), and one IDT involving a mobile wireless station ( $S_2^{M3}$ ). Figure 23 depicts for each message stream, the minimum (MinRT), the mean (MeanRT) and the maximum response time (MaxRT).

The response time of  $S_1^{M1}$  is much higher in the repeater-based scenario than in the bridge-based scenario, since in the second case,  $S_1^{M1}$  benefits from the smaller setting of the  $T_{ID}$  and  $T_{SL}$  parameters (smaller message cycle duration) as well as from the traffic segmentation resulting from the use of bridges (less traffic in  $D^1$ ). It is also noticeable that on the repeater-based scenario there is a sharper increase on

the maximum response time, when the Bit Error Rate (BER) increases. Additionally, the bridge-based approach exhibits a much higher number of concluded transactions (Figure 24), for all message streams.

The response time for  $S_1^{M2}$  is now lower on the repeater-based than on the bridge-based approach. In the bridge-based approach, the response time of an IDT depends on its message stream period. This kind of transaction requires that the initiator performs at least one AL retry before obtaining a response. In this case, the period of message stream  $S_1^{M2}$  is equal to 8 ms, and consequently the MinRT value in the bridge-based scenario must be greater than 8 ms. However, it is noticeable that the MeanRT is very close to the MinRT. This means that there is a very high concentration of response times near the MeanRT.

In relation to  $S_2^{M3}$ , it is noticeable that, on the bridge-based approach, its MeanRT is very close to the MeanRT of  $S_1^{M1}$ , since the stations in these transactions are both mobile stations; therefore most of the response times are obtained when both stations are in the same domain. However, the MaxRT occurs when the stations are on different domains. In these cases, the repeater-based solutions can be more affected by the increase on the BER.

Finally, our simulation results also show that the repeater-based approach did not lose any deadline, while on the bridge-based approach, message streams  $S_2^{M3}$  and  $S_3^{M3}$  lost approximately 0.0001% of the its deadlines. Note that the initiator of these message streams is a wireless mobile station (master M3).

**7.3.4. Frame Length.** Another important factor is to determine which changes on the network configuration affect the network performance; consequently, we decided to change the frame size of message stream  $S_1^{M3}$  to 250 bytes.

On the repeater-based approach there was the need to increase the period to 160 ms and to adjust the  $T_{SL}$ ,  $T_{ID1}$ , and  $T_{ID2}$  parameters. The period for the others messages streams was set using *triang*(140, 160, 180) and the offset was set using *triang*(0, 140, 160). This periodicity causes that the maximum number of transactions in the repeater-based scenario is reduced by 75%. In the bridge-based scenario there was no need to change any network parameter.

Comparing the results depicted in Figure 23 and in Figure 25 we can conclude that increasing the frame length of a message stream has a higher influence on the network performance of the repeater-based scenario.

In the bridge-based scenario, this influence is stronger for message streams which are routed through the same domains as  $S_1^{M3}$ , which is the case of message stream  $S_1^{M2}$ . But for  $S_1^{M3}$  that influence is very small; contrarily, in the repeater-based scenario all message streams are severely affected.

**7.3.5. Error-Prone Environment Conclusions.** In an error-prone environment, our results have showed that the percentage of token losses is similar in both approaches, but the time required to recover from a token loss is much higher on the repeater-based approach due to its parameter settings. Contrarily, the bridge-based solution exhibits a

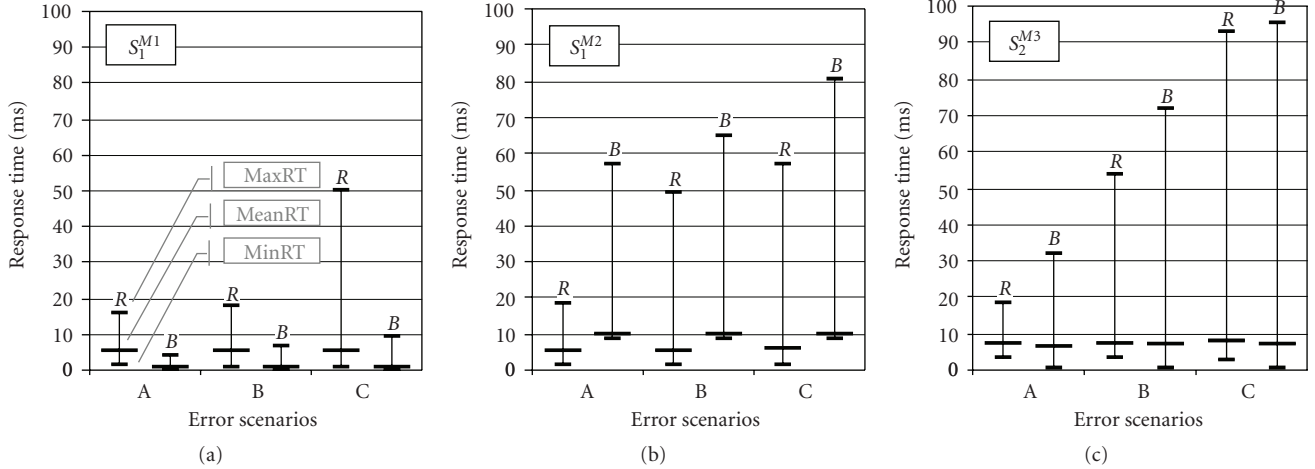


FIGURE 23: Response time.

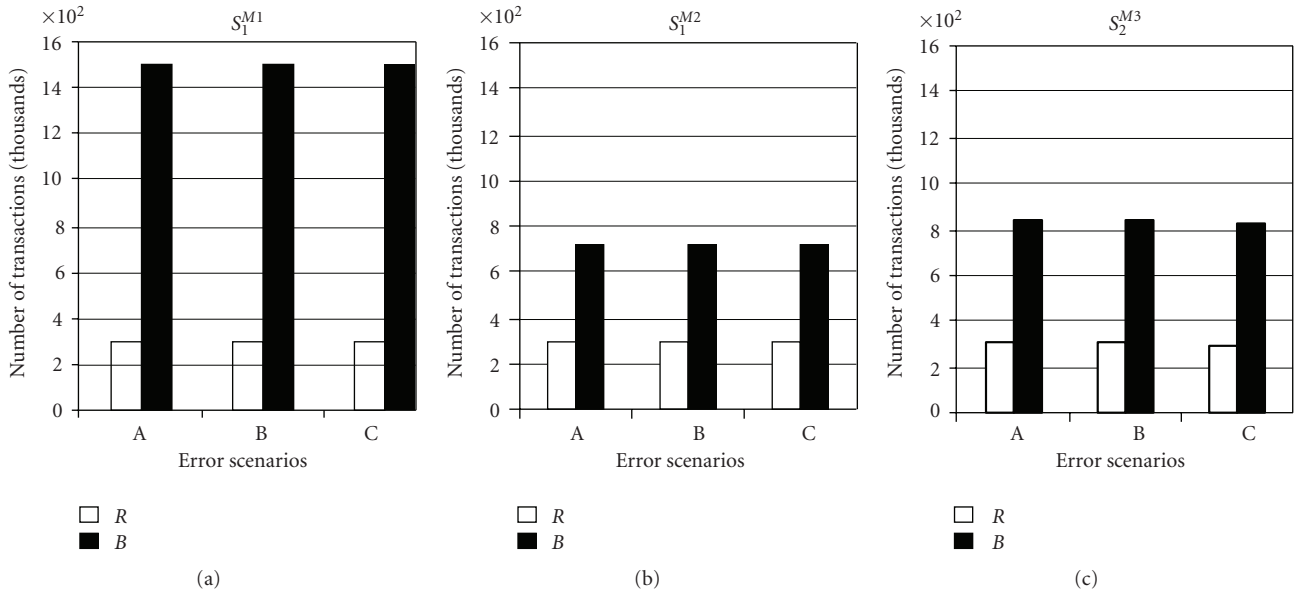


FIGURE 24: Percentage of concluded transactions.

higher percentage of mobility procedure failures due to the complexity of the protocol.

Finally, the bridge-based approach can also achieve a much higher throughput than the repeater-based approach and its performance is less influenced by the increase on the BER and network setting changes.

### 8. Conclusions

In this paper, we have deeply described and analysed a solution in which more than joining technologies it integrates wireless extensions on PROFIBUS networks, connected by Intermediate Systems operating at Data Link layer, as a bridge.

We have shown how simulation tools (i) can help on the validation of new protocols, (ii) provide the means to

compare between different approaches, before developing a real prototype, (iii) compare with analytical results.

We have presented a comparative analysis of the bridge-based solution and another where the Intermediate Systems operate as repeaters (at Physical Layer). In terms of network operation the main difference between the repeater and bridge-based approaches is that the repeater-based approach creates a single broadcast domain and a Single Logical Ring, while the bridge-based approach creates multiple broadcast domains and Multiple Logical Rings (MLRs).

The bridge-based approach benefits from the MLR segmentation, which isolates the traffic between domains permitting lower response times for Intra-Domain Transactions (IADTs). Additionally, the network segmentation permits the independent setting of the network parameters (e.g.,  $T_{ID}$  and  $T_{SL}$ ) in every domain. Contrarily, in the

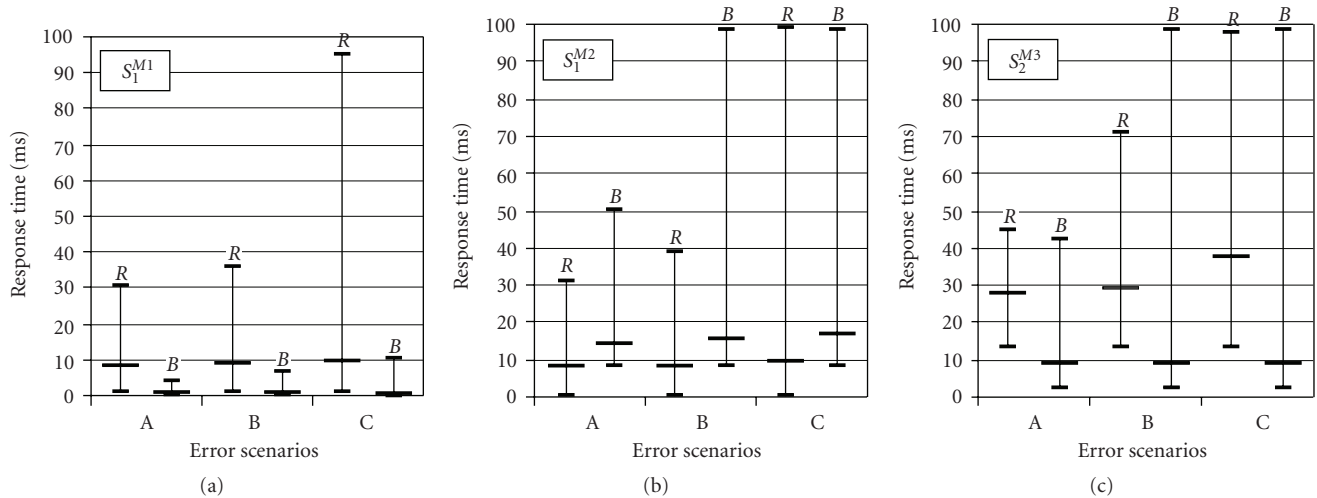


FIGURE 25: Response time according to the frame length.

repeater-based approach, the parameter setting depends on the network parameters and configuration, resulting on higher duration for a message cycle. The segmentation also permits a better responsiveness to errors (transmission and token loss) in the bridge-based approach since  $T_{SL}$  can be set to smaller values. Additionally, the segmentation operated by the bridges permits a higher throughput of the overall network.

From the experiments in which the network parameters and error conditions varied, we concluded that the repeater-based approach is much more influenced by these changes than the bridge-based approach, and in most cases it underperforms the bridge-based approach.

Nevertheless, the bridge-based approach presents some disadvantages when comparing with the repeater-based approach. The repeater-based approach is simpler than the bridge-based approach, since the second requires two new protocols: the Inter-Domain protocol (IDP) and the Inter-Domain Mobility Procedure (IDMP) to be implemented on the Data Link Layer of the bridges. As a consequence of the IDP the response time of the Inter-Domain Transactions (IDTs) depends on the message stream's period and it can be higher than in the repeater-based approach. The IDMP presents a higher level of complexity, which can cause more mobility procedures failures and also higher inaccessibility times, specially, for the wireless mobile stations, but most importantly a bridge-based network recovers from lost token errors much faster.

We had also compared the results from our analytical analysis with the simulation results, which permitted to evaluate the pessimism of the analytical results and we showed how those results can be used for the setting of some network parameters.

Additionally, our work proposes a methodology for the comparison of solutions for industrial wireless networks which can easily be extended to other scenarios. Our work has also been one of the first to propose a hybrid wired/wireless network with real-time and node mobility characteristics for industrial applications.

## Acknowledgments

This work was partially funded by the Portuguese Science and Technology Foundation (Fundação para a Ciência e Tecnologia—FCT) and Instituto Superior de Engenharia do Porto.

## References

- [1] IEC, "IEC61158—Fieldbus Standard for use in Industrial Systems," European Norm, 2000.
- [2] Profinefs, "PROFIBUS, PROFIBUS and PROFINET. PI International," 2008.
- [3] L. Rauchhaupt, "RFieldbus a survey," in *Proceedings of the 5th IFAC International Conference on Fieldbus and Their Applications*, Aveiro, Portugal, 2003.
- [4] RFieldbus, "RFieldbus Manufacturing Field Trial," 2000, <http://www.hurray.isep.ipp.pt/activities/rfpilot/>.
- [5] L. Ferreira, et al., "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*, pp. 283–290, Aveiro, Portugal, 2003.
- [6] L. Ferreira, et al., "Enabling inter-domain transactions in bridge-based hybrid wired/wireless PROFIBUS networks," in *Proceedings of the 9th IEEE International Workshop on Emerging Technologies and Factory Automation*, pp. 15–22, Lisboa, Portugal, 2003.
- [7] L. Ferreira, *A multiple logical ring approach to real-time wireless-enabled PROFIBUS networks*, Ph.D. thesis, University of Porto, Porto, Portugal, 2005.
- [8] K. C. Lee and S. Lee, "Integrated network of PROFIBUS-DP and IEEE 802.11 wireless LAN with hard real-time requirement," in *Proceedings of IEEE International Symposium on Industrial Electronics (ISIE '01)*, vol. 3, pp. 1484–1489, Pusan, South Korea, June 2001.
- [9] A. Willig, *Investigations on MAC and link layer for a wireless PROFIBUS over 802.11*, Ph.D. thesis, University of Berlin, Berlin, Germany, 2002.
- [10] D. Miorandi and S. Vitturi, "Hybrid wired/wireless implementations of Profibus DP: a feasibility study based on Ethernet



- and Bluetooth,” *Computer Communications*, vol. 27, no. 10, pp. 946–960, 2004.
- [11] J. Kjellsson, A. E. Vallestad, R. Steigmann, and D. Dzung, “Integration of a wireless I/O interface for PROFIBUS and PROFINET for factory automation,” *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4279–4287, 2009.
- [12] WirelessHart, “Wireless Hart Specification Released,” IML Group Plc—Control Engineering Europe, 2007.
- [13] P. Ferrari, A. Flammini, and S. Vitturi, “Response times evaluation of PROFINET networks,” in *Proceedings of IEEE International Symposium on Industrial Electronics (ISIE '05)*, vol. 4, pp. 1371–1376, Dubrovnik, Croatia, June 2005.
- [14] Siemens, “Industrial Wireless LAN—I-Features, Applications, Examples,” Automation and Drives Group, 2005, <http://www.automation.siemens.com/>.
- [15] A. Willig and A. Wolisz, “Ring stability of the PROFIBUS token-passing protocol over error-prone links,” *IEEE Transactions on Industrial Electronics*, vol. 48, no. 5, pp. 1025–1033, 2001.
- [16] J. A. Carvalho, A. S. Carvalho, and P. J. Portugal, “Assessment of PROFIBUS networks using a fault injection framework,” in *Proceedings of the 10th IEEE Symposium on Emerging Technologies and Factory Automation (ETFA '05)*, pp. 415–423, Catania, Italy, September 2005.
- [17] M. Alves, *Real-time communications over hybrid wired/wireless PROFIBUS-based networks*, Ph.D. thesis, University of Porto, Porto, Portugal, 2003.
- [18] S. H. Al-Tak, *Design and Simulation of a Transparent Bridge Interconnection for a Fieldbus*, College of Engineering, University of Baghdad, Baghdad, Iraq, 1998.
- [19] L. L. Bello and O. Mirabella, “A multi-ring scheduling strategy for profibus networks,” in *Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society (IECON '01)*, vol. 1, pp. 2144–2148, Denver, Colo, USA, November–December 2001.
- [20] L. Ferreira, et al., “Hybrid wired/wireless PROFIBUS networks supported by bridges/routers,” in *Proceedings of the 4th IEEE International Workshop on Factory Communication Systems*, pp. 193–202, Vasteras, Sweden, 2002.
- [21] M. Alves, et al., “General System Architecture of the RFieldbus Deliverable D1.3,” RFieldbus project IST-1999-11316, 1999.
- [22] P. B. Sousa and L. L. Ferreira, “Bridge Architecture to Interconnect Hybrid Wired/Wireless PROFIBUS Networks,” HURRAY-TR-080102, 2008.
- [23] P. Sousa and L. Ferreira, *Repeater-Based Hybrid Wired/Wireless PROFIBUS Network Simulator*, Polytechnic Institute of Porto, 2006, Hurray-tr-060402.
- [24] J. Banks, et al., *Discrete-Event System Simulation*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2001.
- [25] A. Varga, “OMNeT++ Discrete Event Simulation System,” 2004.
- [26] P. Sousa and L. Ferreira, *Wireless Mobility Simulator*, Polytechnic Institute of Porto, 2006, Hurray-tr-060403.
- [27] T. S. Rappaport, *Wireless Communications. Principles and Practice*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.
- [28] E. Tovar and F. Vasques, “Real-time fieldbus communications using profibus networks,” *IEEE Transactions on Industrial Electronics*, vol. 46, no. 6, pp. 1241–1251, 1999.
- [29] L. Ferreira and E. Tovar, “Timing analysis of inter-cell mobility procedure for wired/wireless PROFIBUS network,” in *Proceedings of the 10th IEEE International Conference on Real-Time and Embedded Computing Systems and Applications*, Gotemburgo, Sweden, 2004.
- [30] S. Behaeghel, et al., “Engineering hybrid wired/wireless fieldbus networks—a case study,” in *Proceedings of the 2nd International Workshop on Real-Time LANs in the Internet Age*, pp. 111–114, Porto, Portugal, 2003.
- [31] A. M. Law and W. D. Kelton, *Simulation Modeling And analysis*, McGraw-Hill, New York, NY, USA, 2000.
- [32] P. Sousa and L. Ferreira, “Repeater vs. bridge-based hybrid wired/wireless PROFIBUS networks: a comparative performance analysis over error prone mediums,” in *Proceedings of the 7th IFAC International Conference on Fieldbuses and nETworks (FET) in Industrial and Embedded Systems*, vol. 7, Toulouse, France, 2007.
- [33] E. Gilbert, “Capacity of a burst-noise channel,” *The Bell System Technical Journal*, vol. 39, pp. 1253–1266, 1960.
- [34] E. Elliot, “Estimates of error rates for codes on burst-nise channels,” *The Bell System Technical Journal*, vol. 42, pp. 1977–1997, 1963.