A Study on the Inaccessibility Characteristics of ISO 8802/5 Token-Ring LANs

INESC Technical Report RT 24-92

J. Rufino, P. Veríssimo

February 1992

A Study on the Inaccessibility Characteristics of ISO 8802/5 Token-Ring LANs

José Rufino, Paulo Veríssimo
Technical University of Lisboa
INESC*
e-mail:...ruf@inesc.pt, paulov@inesc.pt

February 1992

Abstract

Local area networks have long been established as the basis for distributed systems. Continuity of service and bounded and known message delivery latency are requirements of a number of applications, which are imperfectly fulfilled by standard LANs. Most previous studies have addressed this issue by computing worst-case access/transmission delays only for normal LAN operation.

However, LANs are subject to failures, namely partitions. Since most applications can live with temporary glitches in LAN operation, an alternative approach is to quantify all these glitches or temporary partitions, that we named inaccessibility, and derive a worst-case figure, to be added to the worst-case transmission delay in absence of faults. In these conditions, reliable real-time operation is possible on non-replicated LANs. This paper does an exhaustive study of the inaccessibility characteristics of the ISO 8802/5 Token-Ring LAN.

1 Introduction

Local area networks have long been established as the basis for distributed systems. The several variants of standardised LANs (ISO 8802 and FDDI) have different mechanisms to control access to the medium and recover from errors. Continuity of service and determinism in transmission delay are requirements of a number of applications, specially in the fault-tolerance and real-time area, which are unperfectly fulfilled by these LANs, if used without special measures. A number of authors have studied problems such as priority inversion [1], probability of meeting estimated access times [2, 3], extensions for medium failure resiliency through redundancy [4], potential lack of determinism [5].

In reliable real-time systems, the fundamental requirement of communications is that there be a bounded and known message delivery latency, in the presence of disturbing factors such as overload or faults. When the requirement is very strict (eg. for life-critical applications), specialized space-redundant architectures are the solution: point-to-point graphs [6] or multiple LANs [7].

These solutions are however costly and complex. In spite of their limitations, standard LANs are a very important design component. It is worthwhile investigating if the real-time requirement can be reliably met in local area networks that are not replicated, except for eventual medium

*Instituto de Engenharia de Sistemas e Computadores, R. Alves Redol, 9 - 6º - 1000 Lisboa - Portugal, Tel.+351-1-3100000. This work has been supported in part by Junta Nacional de Investigação Científica e Tecnológica (JNICT) through Programa Ciência.

redundancy — at the electrical signalling level. To achieve reliable real-time communication, three fundamental conditions must be validated:

1. bounded delay from request to transmission of a frame[1], given the worst case load conditions assumed;
2. message[2] delivery despite the occurrence of omission failures (eg. lost frames);
3. control of partitions.

Most of the existing studies with this regard have addressed point 1 [2, 8, 9, 10]. However, they are helpless at representing the LAN behaviour, when faults occur. In that case, it is necessary to study the patterns for omission failures (eg. number of consecutive omission failures) and for partitions. Uncontrolled omissions and partitions are a source of asynchrony and inconsistency. This is unacceptable for most systems, let alone real-time ones. Point 2 is addressed under the scope of the LLC type 3 service for point to point interactions. However, it has been practically disregarded for broadcast or multicast interactions. One exception is a modified token-ring mechanism described in[11]. Point 3, when regarding non-replicated networks, and to the best of authors' knowledge, only recently has deserved some attention.

All three points have been addressed in [12] for LANs in general, while point 3 has been specifically addressed in [13] for Token-Bus LANs. A study on the behaviour of an ISO 8802/5 Token-Ring with regard to partitions is the central issue of this paper.

This study contributes to a better understanding of the ISO 8802/5 Token-Ring LAN operation having in mind their reliability and accessibility attributes, issues of mandatory importance for its eventual application in real-time.

2 Controlling Partitions

A network is partitioned when there are subsets of the nodes which cannot communicate with each other[3]. In this sense, a single LAN displays a number of causes for partition, not all of them of physical nature, like bus failure (cable or tap defect): bus contention, ring disruption, transmitter or receiver defects; token loss; etc. Some LANs have means of recovering from some of these situations, and can/should be enhanced to recover from the others, if reliable real-time operation is desired.

However, the recovery process takes time, so in the meantime the LAN is partitioned. A solution to the problem of controlling partitions was presented in [12]. It is based on a very simple idea: if one knows for how long a network is partitioned, and if those periods are acceptably short, real-time operation of the system is possible.

Let us call them periods of *inaccessibility*, to differentiate from classical partitions. The definition of inaccessibility in [14] is summarised here:

> *Certain kinds of components may temporarily* refrain *from providing service, without that having to be necessarily considered a failure. That state is called* **inaccessibility**. *It can be made known to the users of the component; limits are specified (duration, rate); violation of those limits implies permanent failure of the component.*

This is not hard to implement, as shown in [12]. To achieve it one must first assure that all conditions leading to partition are recovered from. For example, tolerance to one medium failure

[1] LAN level information packet.

[2] User level information packet.

[3] The subsets may have a single element. When the network is completely down, *all* partitions have a single element, since each node can communicate with no one.

is assured in the dual FDDI ring [15]. Although the lack of standard redundancy specification, an approach similar to the FDDI ring has been envisaged for the IBM Token-Ring [16].

Then, one needs to show that all the inaccessibility periods are time-bounded and determine the upper bound. The study of ISO 8802/5 Token-Ring inaccessibility is presented next. The scenarios described in the following sections are essentially the inaccessibility periods foreseen in the standard specification. The figures presented illustrate the intervals in the token-ring operation when the LAN does not provide service, although not being failed.

The study yields interesting conclusions, like: a figure for the worst case duration of inaccessibility (to be added to a worst-case access time...); the existence of some periods much longer than others; the reasons that justify those long-term recovery periods.

3 The Token-Ring Network

The ISO 8802/5 Token-Ring is a local area network that uses a token-passing access method over a ring architecture. Individual stations attach to the network through the use of *concentrators*, imposing a *star* topology over the bare ring architecture [17].

Access control uses a token passing protocol. Access to the shared broadcast medium for data transmission is only granted to the station which currently holds the token. A token priority reservation scheme is used to schedule message transmissions accordingly their urgency.

The performance of token-ring access method has been studied along the last decade [18, 19, 20, 21]. Most of these studies assume that the network always operates normally, neglecting the influence of inaccessibility. However, in the presence of faults corrective actions must be performed and in consequence the ring operation is affected, usually severely, in the sense that it can no longer ensure the calculated frame transmission delay bound.

An exploratory analysis on the performance of Token-Ring error handling mechanisms has been provided in [22]. However, it is essentially biased by an experimental evaluation of key error scenarios in a specific network configuration, being hard to infer absolute performance figures in other network configurations.

Our approach is substantially different: we made an exhaustive quantitative analysis of the 8802/5 error handling mechanisms, deriving a set of easy-to-use formulas that can be used to predict the inaccessibility times on any network configuration.

Data Rate ($Mbps$)	Bit Time t_{bit} (μs)	Octect Time t_{oct} (μs)
4	0.25	2.0
16	0.0625	0.5

Table 1: Parameters for an ISO 8802/5 Token-Ring LAN

Network Characterization

For our purposes, two important parameters characterize the basic operation of any ISO 8802/5 Token-Ring network:

- ◇ **Data Rate** - The rate of data signalling, in the ring. It gives a meaning to the *bit* and *octect* times.

◇ **Network Ring Latency** - This parameter accounts the time required for a data unit to propagate once around the ring. It is an aggregate variable accounting for station delays and network size, as expressed by equation:

$$t_{rlat} = t_{PD} + t_{bit} \,.\, (N_{ST} \,.\, lat_{st} + lat_{AM}) \tag{1}$$

- t_{PD} – End-to-end ring cable propagation delay. It is a ring length-dependent variable with a typical value of 5 $\mu s/km$.
- lat_{st} – Station latency, i.e. the delay expressed in bits that each individual station introduces in the circulation of data units.
- lat_{AM} – Depth of the Active Monitor latency buffer, in bits. The extra delay, introduced by this station, aims two goals: assure a minimum latency on the ring that allows token circulation and provide phase jitter compensation [23].
- N_{ST} – Maximum number of stations in the network. Equation (1) defines therefore a worst-case bound for the ring latency instead of their current value. However, notice that its variation can usually be neglected for small variations on the number of ring active stations.

MAC Protocol Data Units

Several types of protocol data frames are exchanged between peer MAC[4] entities to perform the following functions:

– Purge the ring of frame remnants.

– Elect a new active monitor.

– Delimit a failure domain.

– Report the presence of an active monitor on the ring.

– Support of the *neighbor notification process*[5].

– Detect the presence of stations with duplicated addresses.

MAC Frame	Symbol	Duration (μs)	
		$4Mbps$	$16Mbps$
Token	t_{Tk}	6	1.5
Frame header/trailing	t_{HrTr}	42	10.5
Claim Token	t_{CLM}	66	16.5
Duplicate Address Test	t_{DAT}	50	12.5
Active Monitor Alive	t_{AMA}	66	16.5
Standby Monitor Present	t_{SMP}	66	16.5
Beacon	t_{BCN}	70	17.5
Ring Purge	t_{PRG}	66	16.5

Table 2: Duration of MAC Token-Ring Protocol Data Units ($l_{add} = 48$)

The duration of all standard MAC protocol frames which take part in these actions are presented in Table 2, for an address length (l_{add}) of 48 bits. The values were computed based on frame length and data rate.

[4]Medium Access Control.

[5]This process aims to provide each station with the address of its nearest upstream neighbor, required for failure domain delimitation. Additional details on neighbor notification can be found in [24].

MAC Protocol Timers

A set of timers are used by the standard MAC protocol either to detect the abnormal absence of given actions/events or to control the duration/interleaving of others. These timers are listed in Table 3 together with the default values used in the token-ring implementation provided by [23]. Specific timers – *contention monitor* and *unresolved beacon* – used by [23] are also included in the table, although not defined in the standard specification.

MAC Timer	Symbol	Evaluating Expression	Default Values
Return to Repeat	t_{TRR}		4.1 ms
Holding Token	t_{THT}		10 ms
Queue PDU	t_{TQP}		10 ms
Valid Transmission	t_{TVX}	$t_{TRR} + t_{THT}$	10 ms
No Token	t_{TNT}	$t_{TRR} + N_{max} \,.\, t_{THT}$	1 s
Contention Monitor	t_{TCM}		1 s
Active Monitor	t_{TAM}		7 s
Standby Monitor	t_{TSM}		15 s
Unresolved Beacon	t_{TUB}		26 s

Table 3: MAC Protocol Timers

4 Accessibility Constrains

A comprehensive set of scenarios leading to network inaccessibility will be analysed in this section. As a general rule, we start with very simple single error situations that in many cases degenerate into more complex scenarios. Additionally the analysed scenarios progressively evolve to less restrictive – and thus more realistic – operating conditions/fault assumptions. For most of the cases, the main analysis is completely general, being particularised for best and worst cases, afterwards.

Stripping Errors

Before transmitting any queued frame into the network a station must wait for the reception of a *free token*. Should the priority of queued frames be equal or greater than the priority of the received token and they will be elected for transmission. Frame transmission is initiated by setting the *token bit* in the received *access control* field and by clearing the corresponding *monitor* and *priority reservation*[6] bits. The remaining frame control fields, data and ending sequence are appended afterwards.

After completing the transmission of the last queued frame or when there is not enough time remaining in *Timer:Holding Token* (THT) to finish the next transmission, the emitting station will start to send "fill symbols". A new *free token* is generated upon the reception of a frame with a source address matching its own address, but the station will continue to send "fill symbols" until the last transmitted frame is received[7] or until the expiration of *Timer:Return to Repeat* [8]

[6] Frames are always transmitted with the priority of the received token, regardless their own priority.

[7] Identified by the non-assertion of the *intermediate bit* in the *ending delimiter* field.

[8] This timer has been re-started upon transmission of the last frame.

(TRR). In normal operation, this action strips out from the ring all the frames that a station has transmitted. Should the station fail before removing from the ring the transmitted frames, and these would tend to persist in circulating through the ring if appropriate recovery mechanisms were not provided.

A similar error scenario occurs when a station that has issued a *priority token* fails before it is able to restore the former ring service priority, as specified in [24]. In this circumstance, priority tokens would also tend to persist on the ring.

The presence of *circulating frames* or *circulating priority tokens* on the network ring is prevented through a dedicated *active monitor* mechanism. The time required for the elimination of these packets results from the sum of two distinct contributions: the time needed for error detection plus the time required for its recovery, as described by equation (2).

$$t_{ina \leftarrow nostrip} = t_{dect \leftarrow nostrip} + t_{prp} \tag{2}$$

In order to detect the presence of persistent data units on the ring, the active monitor marks every passing frame or priority token, by setting the corresponding *monitor bit*. Any frame or token arriving at the active monitor with its *monitor bit* set is stripped from the ring, by this station.

Error detection can never be performed in a time lower than the network ring latency and its exact value depends on the failed station and active monitor relative locations. In the best-case the active monitor immediately follows the station originating the error whereas in the worst-case the same station immediately precedes the active monitor. This means that both situations differ at most in one ring latency period, as described by equation (3).

$$t_{dect \leftarrow nostrip} = \begin{cases} t_{rlat} & \text{best-case scenario} \\ \\ 2 \,.\, t_{rlat} & \text{worst-case scenario} \end{cases} \tag{3}$$

Upon error detection, the active monitor initiates a recovery procedure that is started with the continuous transmission of *ring_purge* frames. Once the active monitor detects the reception of one of its own *ring_purge* frames, it starts to transmit "fill symbols" during a period given by t_{TRR}, to ensure that all these protocol data units have been stripped from the ring and generates a "new token" afterwards. This *purge ring process* – t_{prp} – terminates timely, in the absence of other network errors, with a best-case duration given by:

$$t^{bc}_{prp} = t_{prPrg} + t_{PRG} + t_{rlat} + t_{TRR} + t_{prTk} \tag{4}$$

where t_{prPrg} and t_{prTk} represent the processing overheads required to generate the *ring_purge* frame and the *token*, respectively. The designation processing overhead, herein employed, is essentially due to a particular token-ring implementation [23] where media access protocols are executed in a special-purpose LAN communications processor. With such an architecture it is reasonable assume that generation of MAC frames and *tokens* may be affected by some latency and therefore we consider worst-case figures, in the processing overheads, for the derivation of an expression that, in the absence of other network errors, gives the worst-case duration of the *purge ring process*:

$$t^{wc}_{prp} = t^{wc}_{prPrg} + t_{PRG} + t_{rlat} + t_{TRR} + t^{wc}_{prTk} \tag{5}$$

Expressions for the best and worst inaccessibility durations, due to the absence of ring stripping can be easily derived. These durations, that we signal with superscripts ^{bc} and ^{wc}, are given by equations (6) and (7).

$$t^{bc}_{ina \leftarrow nostrip} = 2 \,.\, t_{rlat} + t_{prPrg} + t_{PRG} + t_{TRR} + t_{prTk} \tag{6}$$

$$t^{wc}_{ina \leftarrow nostrip} = 3 \,.\, t_{rlat} + t^{wc}_{prPrg} + t_{PRG} + t_{TRR} + t^{wc}_{prTk} \quad (7)$$

Let us now analyse some side effects that token regeneration, by the active monitor, may have on the normal token circulation. Consider the token-ring network pictured in Fig. 1. Station $\mathcal{S}_5$ is either the then-current token holder or the *priority token* remover when it fails, while station $\mathcal{S}_1$ is the then-current active monitor. Station $\mathcal{S}_6$ which was about to receive the token, when the failure of its upstream station, is passed over. So, in addition to the time required for token regeneration, station $\mathcal{S}_6$ must wait an extra token rotation to get access to the network. This means that different stations in the network may have a different view of inaccessibility and that, as a general rule, the inaccessibility time may not be merely represented by the raw inaccessibility duration. It must be consolidated with the addition of the network access delay upper-bound.

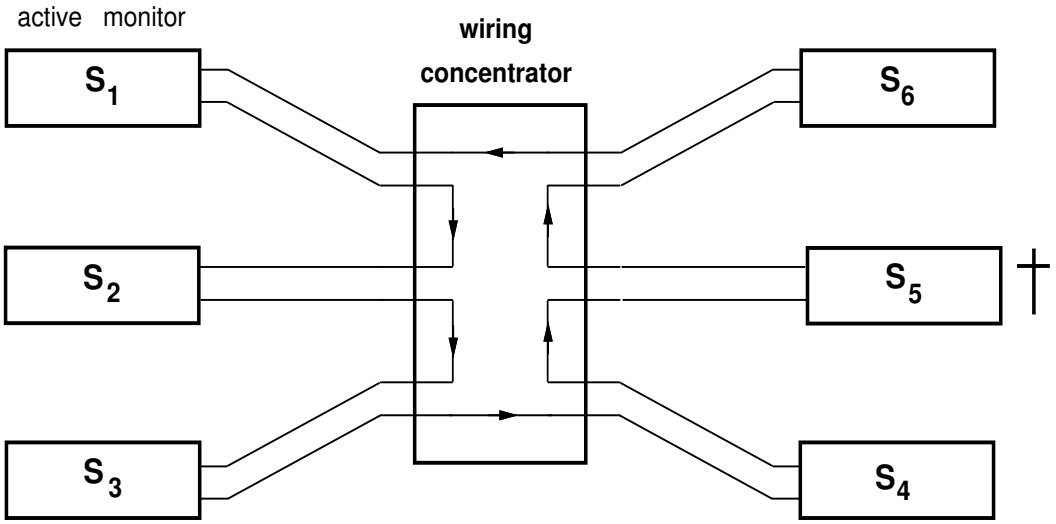

Figure 1: Example of a token-ring network

Bit Corruption Errors

Let us proceed our analysis of token-ring inaccessibility by considering a scenario where some selected bits, inside the frame stream, see their values modified due to the occurrence of faults. Although corruption of individual bits is not likely to happen, this analysis is important because it evidences relevant aspects of protocol robustness.

For example, in 8802/5 frames a set of very important control bits are transmitted outside the FCS coverage domain, due to the need of their "on-fly" modification by the MAC protocol, and it is important to understand how the corruption of these and other equally relevant bits can affect ring operation. This scenario provides a systematic analysis of the possible bit corruption errors.

- **Access Control Field** – The following set of bits are defined within this field:

  - ◇ *Priority and reservation bits* – Corruption of priority and reservation bits can affect both frames and tokens, and they can eventually lead to the issuing of a token with a priority either higher or lower than the correct. In Table 4 we provide a systematic view of the possible error conditions. Errors due to incorrect *priority bits* are more severe than those resulting from the corruption of *reservation bits* since they escape from the control of the *priority reservation* scheme[9]. Its recovery is performed either through

[9] Additional details on priority reservation can be found in [24].

an *active monitor stripping* action or through a dedicated *error recovery procedure* that clears the priority stacks [24]. Errors resulting from reservation bits corruption are recovered strictly by the intervention of the *priority reservation* mechanisms

Token Priority Incorrectness	Corrupted bits	
	Priority	Reservation
Higher	• Incorrect issuing of a priority token. • Elimination by the active monitor.	• Circulation of a priority token. • Reservation of correct priority value. • Correct priority restored on next token release.
Lower	• Incorrect issuing of a low priority token. • **Error Treatment:** clearing of priority stacks.	• Circulation of a low priority token. • Normal priority management and ring utilization operations. • Correct priority restored upon circulation of the low priority token.

Table 4: Consequences of priority/reservation bits corruption

- ◇ *Monitor bit* – This bit is always originally transmitted with a zero value. Its eventual corruption may lead to the premature removal of the frame or priority token from the ring, through the aforementioned *active monitor stripping* action.
- ◇ *Token bit* – The setting of this bit, by corruption, can transform a *free token* into a *busy token* that will be later eliminated through an *active monitor stripping* action. On the other hand, the incorrect clearing of the *token bit* converts a frame into an incorrectly formed token. Such a scenario will be studied ahead.

- **Frame Body** – The frame body is protected by FCS checking and consequently, a corruption in this stream will be detected with the coverage provided by the associated CRC mechanism. Frames with CRC errors will be usually discarded by the MAC sub-layer and, exception made for the eventual corruption of the *source address* field, they do not normally lead to network inaccessibility. Source address recognition is a mandatory condition for token release, by the then-current holding station, and the lack of such event leads to a token loss scenario, to be described ahead. Notice that errors in other frame body fields do not need to be considered since they do not directly affect MAC protocol operation.
- **Ending Delimiter** – A particular bit in the end delimiter — *intermediate bit* — indicates whether or not a given frame is the last transmission issued by the then-current token holding station. Corruption of the *intermediate bit* can lead a station either to the premature or to the late abandonment of *frame stripping* operations. In the first case, frames continue to be repeated by downstream stations, until its elimination by the next token holding station or by the active monitor. On the other hand, the absence of a frame received with its *intermediate bit* cleared maintains the station in "stripping". This situation can be quite complex and calls for a more thorough analysis, to be performed next.

Persistent Stripping

A station that does not recognise its last transmission, due to a corrupted or incorrectly formed ending delimiter, continues to send "fill symbols" and to strip any arriving data, until the expiration of *Timer:Return to Repeat*. Assuming that no other errors have occurred, the token should be

released by the stripping station and it may be used then by another station. In this scenario, three distinct situations may happen:

- No station uses the token, that will be eliminated by the *stripping station*, thus leading to a token loss scenario.
- The next station using the token transmits during a period – t_{tx} – obeying to relation $t_{tx} \leq t_{TRR}$. Due to the presence of the *stripping station*, no frames will be repeated downstream until detection of the then-current token holder last transmission[10] or until a period given by t_{TRR} has elapsed. In either cases, the token holding station does not hear any of its own transmissions and consequently, it will not release the token. Such a situation degenerates therefore in a token loss scenario.
- The station using the token transmits during a period given by relation $t_{tx} = t_{TRR} + t_{ltx} \leq t_{THT}$, where t_{ltx} is the duration of the last $n \geq 1$ transmissions. In the absence of other errors, these n transmissions are correctly repeated by the former *stripping station* and received by the then-current token holding station. This allows the straight restoration of normal ring operation.

The presence of a *stripping station* prevents any frame repetition and consequently their reception, during a time interval bounded by t_{TRR}, in all the downstream stations till the station retaining the token (the out-partition). At least during this period the network is partitioned, meaning that only stations located between the then-current token holder and the *stripping station* (the in-partition) will be able to receive that station transmissions. This selective partition pattern, when strictly due to MAC protocol operation, is typical of networks with a ring topology. During the inaccessibility period, at most one station – defining the in-partition start boundary – can capture the token. A second network access, by this station, will only be possible after recovery. This means that inaccessibility due to the presence of a *persistent stripping* station is controlled both in terms of their number of occurrences, as explained above, and duration, with best and worst-case figures given by:

$$t^{bc}_{ina \leftarrow pstrip} = t_{TRR} \tag{8}$$

$$t^{wc}_{ina \leftarrow pstrip} = t^{wc}_{ina \leftarrow tkloss} \tag{9}$$

Bad Format Errors

In a previous scenario we have analysed the impact of individual bit corruption on MAC protocol operation. However, individual bit corruption seldom occurs; extensive adulteration of frame fields due to corruption bursts is a more realistic scenario. On the other hand bit modification is not the only source for frame errors: EMI can originate electric signaling code violations and frames can be truncated by a deficient operation of physical layer components.

Let us group these different source of errors into the unique designation of bad format errors. Two particular cases are considered: start of frame sequence and badly formed tokens. The remaining cases either were already addressed in the previous scenarios or are not relevant for our study of MAC protocol robustness.

- **Start Frame Sequence** – If the start of frame sequence, i.e. either the *start delimiter* or the *access control* fields are incorrectly formed, no frames or tokens will not be recognised by network stations. In consequence, the scenario degenerates in the token loss, to be described ahead.
- **Bad Token Format** – In this case we analyse the behaviour of token-ring stations upon reception of a token whose start sequence (SD and AC fields) is not immediately followed by a correct ending delimiter. This can result from the transformation of a frame into a token through the modification of its *token bit* or due to a corrupted/truncated ending delimiter.

[10] Detection of a frame with the *intermediate bit* cleared is a sufficient condition for abandon stripping operations.

Let us assume that a station in the network captures such a token and starts using it by modifying the corresponding *token bit*. Once it detects that the token is not properly formed, it immediately terminates frame transmission with the issuing of an abort sequence and makes a transition to the *repeat state* [24]. The frame remnant, previously transmitted, will be soon eliminated through an *active monitor stripping* action. An incorrect formed *priority token* will be eliminated in the same way, even if it is not captured by any station. On the other hand, an incorrect *non-priority token* that no station tries to use will tend to persist in circulating through the ring. This error is usually also corrected by the active monitor, but the exact set of actions rather depends of each particular system architecture.

For example, [22] mentiones that in the IBM Token-Ring, the reception of a correct start sequence is a sufficient condition for token recognition. Therefore, no recovery action will be initiated in this case and the defective *non-priority token* remains undetected, continuing to circulate on the ring, until a station tries to use it. Conversely, in other implementations a token might be only recognised when properly formed. In consequence, recovery from a token format error can be triggered either by an attempt to use the defective token or by an abnormal absence of valid packets on the ring.

In this study, we will consider a network operating accordingly with this last approach. This favours network performability in situations of low network traffic. However, the two methodologies provide nearly identical results in situations of fair to high traffic loads.

Token Loss

The loss of a token can be due either to the failure of the then-current token-holding station or to the aforementioned errors in network operation (i.e. unrecognisable start of frame sequence, source address field, last transmission indicator or bad token format).

Token losses are detected by monitoring the absence of correct ring activity and recovered through the already described *purge ring process*. The whole process is leadered by the then-current active monitor. Each time that a frame or a token passes through the active monitor a dedicated MAC protocol timer that monitors valid network transmissions is re-started. In the absence of valid formed packets *Timer:Valid Transmission* (TVX) is not re-started[11] and will expire. This event triggers the process of purging the ring from any frame remnant and the subsequent issuing of a new token. The duration of the resulting inaccessibility period is given by:

$$t_{ina \leftarrow tkloss} = t_{dect \leftarrow tkloss} + t_{prp} \tag{10}$$

where $t_{dect \leftarrow tkloss}$ is the token loss error detection latency and t_{prp} is the duration of the *purge ring process*. The time needed for error detection depends on the time elapsed between the last TVX re-start and the occurrence of the fault. In the worst-case the fault occurs just after TVX re-start, while in the best-case the token almost completes an entire round before its loss. Therefore, the corresponding error detection latency varies between the nominal value of TVX minus one ring latency period and their exact value.

$$t_{dect \leftarrow tkloss} = \begin{cases} t_{TVX} - t_{rlat} & \text{best-case scenario} \\ \\ t_{TVX} & \text{worst-case scenario} \end{cases} \tag{11}$$

The conjunction of this expression with equations (4) and (5) allow us to derive the best and worst-case bounds for this scenario.

[11] As a matter of fact, only *non-priority tokens* need to be properly formed. Incorrect frames and priority tokens can always re-start TVX timer, provided that their start sequences are valid formed. This procedure aims to speed-up error recovery by allowing token regeneration to be carried out through an *active monitor stripping* action rather than by token loss detection.

$$t^{bc}_{ina\leftarrow tkloss} = t_{TVX} + t_{prPrg} + t_{PRG} + t_{TRR} + t_{prTk} \tag{12}$$

$$t^{wc}_{ina\leftarrow tkloss} = t_{TVX} + t^{wc}_{prPrg} + t_{PRG} + t_{rlat} + t_{TRR} + t^{wc}_{prTk} \tag{13}$$

Active Monitor Loss

In the presence of a long term ring error condition, the previous method for token regeneration may not timely terminate. The duration of the *purge ring process* is controlled, inside the active monitor, through the use of *Timer:No Token* (TNT), which is started on process initiation. Upon its timeout, due to the non-receiving of its own *ring_purge* frames, the active monitor abandons its protocol functions to enter in *standby monitor* operation. However, this is not the only cause for active monitor loss. For example, its physical layer components may not provide a properly operating *latency buffer* or a correct *master ring clock* and even the whole station may fail.

In all the above situations, the structure supporting token circulation collapses. This event is detected in the *standby monitors* through the use of a dedicated mechanism based on *Timer:No Token* (TNT) which, under normal operation, is re-started upon each token passing[12]. In the absence of such data units, TNT is not re-started and will expire, triggering then the election of a new active monitor.

A more complex active monitor loss situation occurs when its MAC sub-layer, due to a failure, stops to ensure the corresponding protocol functions, i.e. circulating packets checking, purge ring control and token generation. Should these mechanisms be required for the recovery of other ring errors and the situation will be detected in some *standby monitor*, upon TNT timeout. However, in the absence of those errors, a situation where MAC active monitor functions are not assured, by any station in the network, might subsist for a very long period. In order to place an upper bound on this time, the active monitor activity is tracked by the remaining network stations. While operating correctly, the active monitor cyclely enqueues for transmission, with a period given by *Timer:Active Monitor* (TAM), an *active_monitor_alive* frame that, when received by the standby monitors, re-starts their *Timer:Standby Monitor* (TSM). In the absence of these active monitor indications, some TSM timers will eventually expire[13]. Stations where this event occurs, start the election of a new active monitor.

Therefore, the standby monitors are provided with two distinct mechanisms for the detection of active monitor failures. Since both are timer-based, the error detection latency actually depends on the time elapsed between the last timer re-start and the occurrence of the fault. The method based on the TNT aims to speed-up error detection upon the failure of physical layer components[14] while the active monitor tracking covers MAC sub-layer failures[15]. The resulting error detection latency is given by expression (14).

$$t_{dect\leftarrow AMloss} = \begin{cases} t_{TNT} - t_{rlat} & \text{best-case scenario} \\ \\ t_{TSM} & \text{worst-case scenario} \end{cases} \tag{14}$$

[12]This mechanism is bypassed under ring recovery by allowing timer re-start to be performed also upon reception of *beacon*, *claim_token* or *ring_purge* frames.

[13]Active monitor tracking is bypassed under ring recovery, by allowing TSM re-start upon reception of *beacon* frames.

[14]This is the standard detection methodology, with best and worst cases given by $(t_{TNT} - t_{rlat})$ and (t_{TNT}). Nevertheless, a subset of severe ring faults can be detected more efficiently by a method that we will describe in the next scenario.

[15]In a time between $(t_{TSM} - t_{TAM})$ and (t_{TSM}).

For the evaluation of the corresponding inaccessibility period, two distinct contributions must be added to the error detection latency: the time required for the election of a new active monitor and the time spent in the token regeneration.

$$t_{ina \leftarrow AMloss} = t_{dect \leftarrow AMloss} + t_{tcp} + t_{prp} \tag{15}$$

The exact duration of a new active monitor election, to be performed through a *token claim process* – t_{tcp} – depends on whether or not a contention between stations occurs and on how quickly this contention is resolved. The *token claim process* starts with the transition of a standby monitor from the *standby* to the *transmit claim* state, where *claim_token* frames are continuously transmitted until they are received back from the network. Upon this event a transition is made to the *active monitor state* [24], initiating then the aforementioned *purge ring process*. In the best-case only one standby monitor executes token claiming. No contention occurs and assuming that the first transmitted *claim token* frame is correctly received, the duration of the process is given by equation:

$$t_{tcp}^{bc} = t_{prClm} + t_{CLM} + t_{rlat} + t_{prAM} + t_{ctc} \tag{16}$$

where t_{prClm} and t_{prAM} are the overheads required for the processing of *transmit claim* and *active monitor* transitions, respectively. The time accounted by t_{ctc} – claiming time correction – represents a corrective term that accommodates modeling of some token-ring implementations [23] where reception of more than one *claim_token* frame is required for process termination. Such corrective term is given by equation:

$$t_{ctc} = N_{Clm} \,.\, (t_{CLM} + t_{CIFS}) \tag{17}$$

where N_{Clm} is the number of extra *claim_token* receptions and t_{CIFS} is the inter-frame spacing between successive *claim_token* frames.

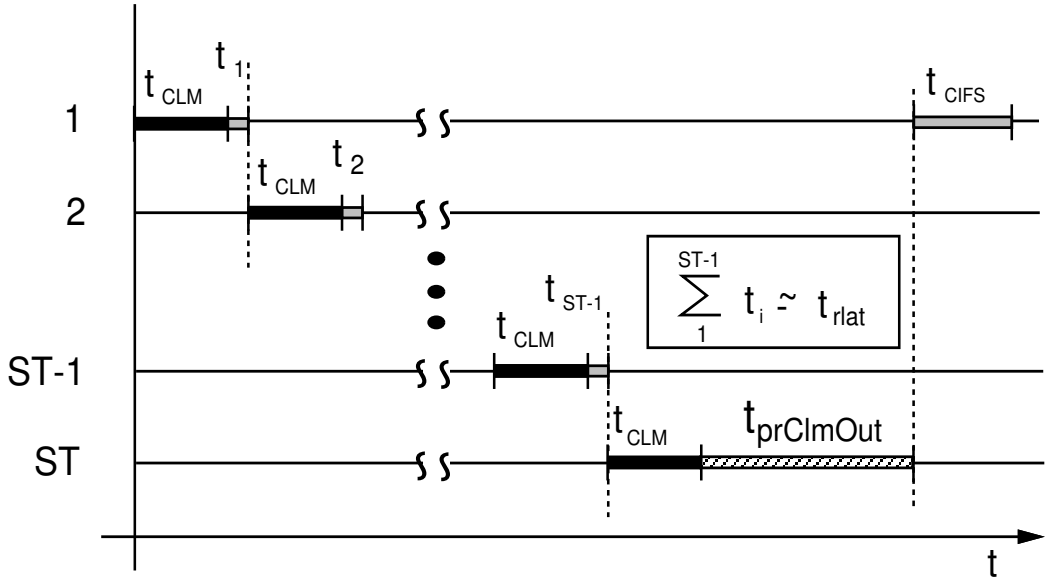

Figure 2: Contention resolution on token claim process (worst-case scenario)

Conversely, a contention process must be resolved when there are several stations in the *transmit claim* state. The resolution algorithm relies on station addresses discrimination: stations receiving a *claim_token* frame with a source address higher than their own, from upstream stations, abandon the process by making a transition into the *standby* state [24]. A worst-case resolution scenario is

represented in Figure 2, where the duration of each *claim_token* frame is denoted by t_{CLM} and its propagation delay to the nearest downstream neighbor is identified by t_i. The process is leadered by the station having the highest address and we assume that all downstream stations marginally miss the *claim_token* transmission of its upstream neighbor, initiating then the execution of the corresponding *token claim process*. Furthermore we assume that *claim_token* transmissions cannot be preempted and that the exit from this state takes $t_{prClmOut}$. The duration of the *resolve contention process* is therefore upper-bounded by equation:

$$t_{rcp}^{wc} = N_{ST} \,.\, t_{CLM} + t_{rlat} + t_{prClmOut}^{wc} + t_{CIFS} \tag{18}$$

that added to the time spent in the remaining token claiming actions, provides a worst-case bound for the duration of the *token claim process*, in the absence of other network errors.

$$t_{tcp}^{wc} = t_{prClm}^{wc} + t_{CLM} + t_{rlat} + t_{prAM}^{wc} + t_{ctc} + t_{rcp}^{wc} \tag{19}$$

Signal Loss

In the previous scenario we have described the bare mechanism, forecasted in the standard specification, for the detection of faulty physical layer operation. In addition to that mechanism, some token-ring implementations [23] present special provisions for the detection of severe physical layer errors, like an input signal with insufficient energy or grossly out of phase.

First, a station detecting the reception of five consecutive half-bits of Manchester coded data without a phase change, begins to transmit "fill symbols" until it detects a valid signal. This prevents other stations of detecting the error. Secondly, a recovery procedure is entered if the error condition persists beyond a time period given by $t_{dect \leftarrow signloss} < t_{TNT}$. Usually, the station detecting the error directly enters into the *standby monitor: transmit claim* state, unless it is under a strongest recovery action or in the course of a *ring insertion process*.

This procedure can significantly improve network performability: not only reduces the error detection latency but also creates conditions to allow recovery actions to be started in isolation and, probably, finished without contention.

Ring Interruption

The loss of a station input signal is usually a consequence of physical ring interruption, either due to normal operational constrains or due to the failure of its physical layer components. Assuming a worst-case viewpoint, we will perform, in this scenario, a systematic analysis of data flow interruption that only takes into account their duration. The relation of this error analysis with their physical causes will be performed in subsequent scenarios.

Let us, therefore, assume that the flow of data in the physical ring is interrupted during a time given by t_{RgErr}. The set of procedures, to be executed on error recovery, rather depends on the ring interruption duration, as follows:

i) $t_{RgErr} < t_{Purge}$ — where t_{Purge} is given by equation (11). The impact of such a short-term error condition depends on the extension of corrupted data. Most likely, a worst-case scenario where the token is destroyed occurs, with its regeneration being carried out through the active monitor, as formerly explained.

ii) $t_{Purge} \leq t_{RgErr} < t_{Claim}$ — As we have described, the active monitor initiates, upon TVX time-out, the execution of a *purge ring process*. Whenever ring integrity is restored before the entering of some station into the *token claiming* state, this process allows a straight token regeneration. The deadline for purge ring termination, given by (20), actually depends on whether or not the

station processing the error is provided with the aforementioned dedicated mechanism for signal loss detection.

$$t_{Claim} = \begin{cases} t_{dect \leftarrow signloss} & \text{signal loss detection} \\ t_{TNT} - t_{rlat} & \text{no signal loss detection} \end{cases} \tag{20}$$

The resulting inaccessibility period is therefore given by:

$$t_{ina \leftarrow noring}(noClaim) = t_{RgErr} + t_{prp} \tag{21}$$

iii) $t_{Claim} \leq t_{RgErr} < t_{Beacon}$ — When the conditions described in the previous items are not met, some station in the network will initiate a *token claim process* for the election of a new active monitor. Provided that ring integrity is restored in a time that allows process termination, the resulting inaccessibility period will be given by:

$$t_{ina \leftarrow noring}(noBeacon) = t_{RgErr} + t_{tcp} + t_{prp} \tag{22}$$

with the upper-bound for error duration given by the sum of the ring error detection latency with the token claim process marginal termination time $t_{tcp}^{mt} = t_{TCM}$.

$$t_{Beacon} = t_{Claim} + t_{tcp}^{mt} \tag{23}$$

iv) $t_{Beacon} \leq t_{RgErr}$ — If the ring error condition persists beyond t_{Beacon} a recovery mechanism stronger than the *token claim process* is entered. The *beacon recovery process* is therefore only started under severe ring fault conditions. The process is initiated by repeatedly transmitting a *beacon* frame which carries information about the detected error (e.g. signal loss, TCM timeout) and the most reliable upstream neighbor address [24]. Stations in the *beacon* state, receiving *beacon* frames issued by other upstream stations abandon beaconing by making a transition into the *standby* state. At last, only the station located immediately downstream to the ring interruption remains beaconing, thus allowing the delimitation of the failure domain. Restoration of ring physical integrity usually requires network reconfiguration. The exact set of actions, to be performed for ring recovery, rather depends of each particular case and will be discussed in next scenarios. For now, we simply assume that the *beacon recovery process* timely terminates, upon reception of its own *beacon* frames, with a resolution period given by t_{brp}. The *beacon recovery process* is followed by the election of a new active monitor and by the corresponding token issuing.

$$t_{ina \leftarrow noring}(Beacon) = t_{Beacon} + t_{brp} + t_{tcp} + t_{prp} \tag{24}$$

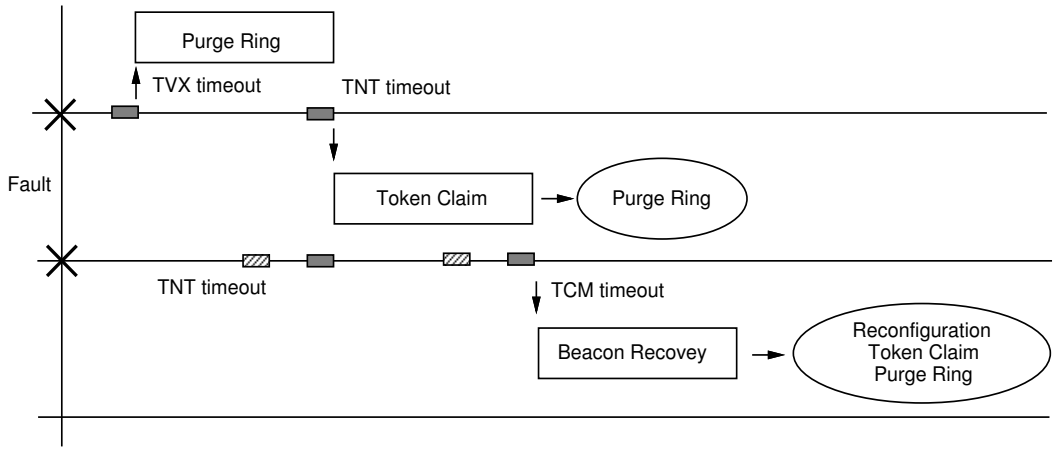

Figure 3: Token-ring recovery processes

Dumb Transmitter

Let us assume that a given station fails in the sense that it stops to transmit/repeat network data. Due to the permanent nature of the failure, network reconfiguration is required. This action is performed as a consequence of a *beacon recovery process* execution: a station receiving a *beacon* frame whose *upstream neighbor address* field matches the station's physical address, assumes that its transmitter is failed and removes it-self from the physical ring to execute a set of *confidence tests*. Since the station's transmitter is actually failed, the station will not re-insert.

The *beacon recovery process* is always wined by the nearest downstream neighbor of the failed station. Assuming that no other station, but the process winner issues *beacon* frames, no contention occurs and the propagation time of these data units, till the station with the failed transmitter, will be simply given by the network ring latency:

$$t_{bpd}^{bc} = t_{BCN} + t_{rlat} \tag{25}$$

In a less favorable scenario other stations in the network may also enter in this process and the *beacon propagation delay* – t_{bpd} – will be more long, as a consequence of station contention. For the computation of a worst-case bound, we consider that all downstream stations marginally miss the reception of its upstream neighbor *beacon* frame, initiating then its own *beacon* transmission. Furthermore, we assume that this transmission cannot be preempted and that the exit from this state takes $t_{prBcnOut}$. The time required for the propagation to the dumb station of a *beacon* frame issued by its nearest downstream neighbor is, under these circumstances, given by:

$$t_{bpd}^{wc} = N_{ST} \,.\, t_{BCN} + t_{rlat} + t_{prBcnOut}^{wc} \tag{26}$$

where t_{BIFS} is the inter-frame spacing between successive *beacon* frames.

The resulting inaccessibility period is obtained from equation (24) using a resolution period given by equation:

$$t_{brp\leftarrow dtx} = t_{prBcn} + t_{bpd} + t_{st\leftarrow leave} + t_{BIFS} + t_{BCN} + t_{btc} + t_{rlat} \tag{27}$$

where t_{prBcn} is the overhead required for entering into the beacon state and $t_{st\leftarrow leave}$ is the time required for switching the station's attaching relay from the *inserted* to the *bypassed* state. Notice that after dumb station removal, it is required that a beacon frame must be received back, in the beaconing station, to allow the *beacon recovery process* to terminate. The time accounted by t_{btc} – beaconing time correction – represents a corrective term that accommodates modeling of some token-ring implementations [23], where reception of more than one *beacon* frame is required for process termination. Denoting by N_{Bcn} the number of extra *beacon* receptions, such corrective term is given by:

$$t_{btc} = N_{Bcn} \,.\, (t_{BCN} + t_{BIFS}) \tag{28}$$

This description essentially follows the features of the token-ring implementation provided by [23], particularly in those issues concerning signal loss detection – as a result of the dumb transmitter – and station auto-removal for confidence testing. Similar considerations also apply to the next scenario.

Deaf Receiver

If the ring interruption is due to the failure of the beaconing station receiver, the auto-removal of its upstream neighbor does not restore physical ring integrity. As a matter of fact, after termination

of their *confidence tests* that station re-inserts in the ring, since they will not detect any failed component. The station with the failed receiver will remain transmitting *beacon* frames until expiration of *Timer:Unresolved Beacon* (TUB), as implemented in [23]. At that time, the station with the failed receiver de-inserts from the ring and their transmission of *beacon* frames stops.

Usually, the time required to complete station auto-removal – $t_{st \leftarrow leave}$ – is much greater than the network ring latency – t_{rlat} – and therefore, any *beacon* frame that may be in repetition throughout the ring is, in principle, destroyed during reconfiguration. If this is not the case, circulation of a beacon frame issued by a station that does not belong to the ring anymore, will entirely block its operation, since it prevents the initiation of any other recovery action. To cope with this situation, the existing mechanism supporting detection of circulating packets is activated in a given station, upon reception of two consecutive *beacon* frames originated from its nearest upstream neighbor. If a *circulating beacon* frame is detected by this station a transition to the *transmit claim* state is made.

In a more realistic situation, no *beacon* frames subsist on the ring after beaconing station auto-removal, and a dedicated mechanism is provided for the detection of an abnormal absence of *beacon* frames during the execution of a *beacon recovery process*. If no *beacon* frames are received for a period of $t_{dect \leftarrow noBcn}$, it is assumed that the conditions causing the ring interruption have been corrected and a transition is made to the *transmit claim* state.

This is the case considered in our analysis. The inaccessibility period, due to the presence of a deaf receiver, is given by equation (24), with a resolution period given by equation (29).

$$t_{brp \leftarrow drx} = t_{prBcn} + t_{TUB} + t_{st \leftarrow leave} + t_{dect \leftarrow noBcn} \tag{29}$$

Streaming Receiver

In this scenario, we consider that the receiver section of a given station, although recognising the presence of a correct input signal, fails in the provision of correctly decoded data at the PHY-MAC interface. Essentially, recovery from this error follows the steps used to bypass a deaf receiver, with the differences listed below:

- Error detection is performed by TNT timeout instead of by signal loss detection.
- Since TNT can expire in several stations, contention for the establishment of a *beacon recovery process* can occur. Beacon frames are transmitted either with a *received claim token* or *no received claim token* indications.
- Only the faulty station de-inserts from the ring, upon TUB timeout.

The resulting inaccessibility period will be expressed by equation (24), with a resolution period given by (29) and a t_{Beacon} value defined by:

$$t_{Beacon} = t_{TNT} - t_{rlat} + t_{TCM} \tag{30}$$

Jabbering Transmitter

Let us assume a token-ring implementation that, as in [23], is provided with a watchdog-timer monitoring station insertion. A reset of this timer must be performed, periodically, by the MAC protocol to ensure that the station remains inserted.

Additionally, let us assume that the MAC protocol functions stuck in the continuous transmission of frames, tokens or abort sequences and that, in consequence, it can no long ensure the timely reset of the *physical insertion watchdog-timer*. In these conditions, a jabbering station will de-insert

from the ring, in a time given by t_{PhyOut}. The duration of the ring interruption is therefore given by:

$$t_{RgErr \leftarrow jabber} = t_{PhyOut} + t_{st \leftarrow leave} \tag{31}$$

The resulting inaccessibility period is given accordingly one of the expressions (21) through (24).

Station Insertion

Stations are physically inserted in the network ring through the switching of a dedicated relay from the *bypass* to the *inserted* state. In the course of this action, the network ring is interrupted during the period required for relay switching and bounce elimination. The duration of this disturbance, herein denoted by $t_{st \leftarrow join}$, is usually small but can, nevertheless, interrupt the normal token flow, due to its corruption. Assuming that $t_{st \leftarrow join} < t_{TVX}$, the resulting inaccessibility period will be equivalent to a token loss scenario.

Multiple Insertions

In the best-case all the stations will simultaneously switch their *attaching relays*, and the situation will not be distinguished from the insertion of a single station. In the worst-case, station insertions are serialised. An upper-bound for ring interruption duration can be obtained assuming a massive join to a network previously formed only by two stations.

$$t_{RgErr \leftarrow mjoin} = (N_{ST} - 2) \, . \, t_{st \leftarrow join} \tag{32}$$

The resulting inaccessibility period is either given by equation (10) or accordingly one of the expressions (21) through (24).

Station De-Insertion

Ring interruption also occurs when a station switches its *attaching relay* from the *inserted* to the *bypass* state. The duration of ring interruption is denoted by $t_{st \leftarrow leave}$ and accounts the time required for relay switching and bounce elimination.

Multiple De-Insertions

In the best-case, all the stations simultaneously de-insert from the ring and the situation cannot be distinguished from the withdraw of a single station. In the worst-case, these events are serialised and upon a massive withdraw of $(N_{ST} - 2)$ stations, the duration of ring interruption is given by:

$$t_{RgErr \leftarrow mleave} = (N_{ST} - 2) \, . \, t_{st \leftarrow leave} \tag{33}$$

Broken Lobe

Upon an abrupt breakage of a station's lobe connection, a total loss of the *DC phantom drive current* is detected and the station's *attaching relay* is disconnected, in a period given by $t_{st \leftarrow leave}$. More subtle faults, originated by the interruption of only one line in the transmit/receive pair and that can cause an abnormal *binary error rate* in this link, are detected by monitoring the balance of the DC current. If an unbalanced drive current persists beyond a period given by $t_{WireFault}$, the station removes it-self from the ring. The corresponding ring interruption period is thus given by:

$$t_{RgErr \leftarrow brklobe} = \begin{cases} t_{st \leftarrow leave} & \text{best-case scenario} \\ t_{WireFault} + t_{st \leftarrow leave} & \text{worst-case scenario} \end{cases} \quad (34)$$

Broken Backbone

In the presence of a broken backbone the auto-removal of those stations located in the nearest neighborhood of the failure domain[16], performed in an attempt to restore ring integrity, will not succeed. These stations will re-insert on the ring, after completion of their confidence tests and we return back to the situation existing prior to station's auto-removal.

Using today's technology, restoration of ring integrity only occurs after manual reconfiguration. Such situation is clearly unacceptable for real-time applications. The study of automatic reconfiguration mechanisms able to recover from these faults fall out from the scope of this work, but are of mandatory importance for an ISO 8802/5 LAN with enhanced availability.

Data Rate - $4Mbps$		
Scenario	t_{ina} (ms)	
	min.	*max.*
Stripping Errors	4.6	4.7
Persistent Stripping	4.1	14.7
Token Loss	14.6	14.7
Active Monitor Loss	1044.9	15068.4
Dumb Transmitter	1416.0	1441.7
Deaf Receiver	27455.2	27478.3
Streaming Receiver	28255.2	28278.3
Jabbering Transmitter	23.3	23.4
Station Insertion	14.6	14.7
Multiple Insertions ($N_{ST} = 32$)	14.6	154.7
Station De-Insertion	14.6	14.7
Multiple De-Insertions ($N_{ST} = 32$)	345.2	368.3
Broken Lobe	14.6	5078.4

Table 5: Token-Ring Inaccessibility Times ($l_{add} = 48$)

Analytic Results

In order to complete our study of networking accessibility we now evaluate the inaccessibility time bounds, for a given token-ring network. The network length is $C_l = 500m$ and the total number of stations $N_{ST} = 32$. We have assigned a typical value of $200\mu s$ to all the processing times, with an overhead of 25% for their worst-case figures. These values are estimates, supported by our personal experience on the evaluation of communication protocols performance. However, an experimental measurement of these parameters is a mandatory condition when a highest degree of accuracy is required. To all the remaining implementation dependent parameters we have assigned the set of values which characterise the behaviour of [23] ($lat_{AM} = 27$, $lat_{st} = 2.5$, $N_{Clm} = 2$, $N_{Bcn} = 7$, $t_{CIFS} = t_{BIFS} = 20ms$, $t_{dect \leftarrow signloss} = 200ms$, $t_{PhyOut} = 8.7ms$ and $t_{WireFault} = 5s$). Further

[16] Upstream and downstream.

estimates, which are essentially dependent on the performance of physical layer components, are: $t_{st \leftarrow join} = 5ms$ and $t_{st \leftarrow leave} = 10ms$. The results of the evaluation, for all the relevant studied scenarios, are summarized in Table 5.

The worst-case figures are extremely high, like for example the 15 seconds that may take the establishment of a new active monitor and the 28 seconds required for the shutdown of a streaming receiver. However, this study not only provides a basis to know what to expect from token-ring performance in the presence of failures, but also provides guidance on where to act in order to improve the situation and *justifiably* achieve better performability, and thus better respect any bounded delay requirements.

With these grounds, we can recognise that longest inaccessibility periods are essentially due to an over dimensioning of the standard MAC timers. The definition of inaccessibility minimizing strategies falls out from the scope of this work. Nevertheless, we can envisage that a realistic tuning of MAC protocol timers with each specific network configuration can optimise error-handling performance and bring the inaccessibility figures drastically down. This will be a mandatory condition for the eventual utilization of ISO 8802/5 Token-Ring LANs in real-time applications.

5 Conclusions

To achieve reliable real-time operation of a local area network, a bounded delay requirement must be met. Most previous studies have addressed this issue by computing worst-case access/transmission delays only for normal LAN operation.

However, achieving the bounded delay requirement means, amongst other factors, ensuring continuity of service. LANs are subject to failures, namely partitions: if these are not controlled, the above mentioned requirement is not met. While LAN replication is a solution, it is costly and complex. Some applications can live with temporary glitches in LAN operation, so an alternative approach is to quantify all these glitches or temporary partitions, which we have named *inaccessibility* periods, and derive a worst-case figure, to be added to the worst-case transmission delay expected in the absence of faults.

In these conditions, reliable real-time operation is possible on non-replicated LANs, through appropriate techniques[12].

This paper does an exhaustive study of the inaccessibility characteristics of the ISO 8802/5 Token-Ring LAN, addressing the problem of temporary LAN partitioning in a comprehensive way. The derived error-handling performance model allows the evaluation of corrective terms for computing transmission delays in various situations.

References

[1] Jeffery H. Peden and Alfred C. Weaver. The utilization of priorities on token ring networks. In *Proceedings of the 13th Conference on Local Computer Networks*, Minneapolis, USA, October 1988.

[2] D. Janetzky and K. Watson. Token bus performance in MAP and PROWAY. In *Proceedings of the IFAC Workshop on Distributed Computer Protocol System*, 1986.

[3] Marjory J. Johnson. Proof that timing requirements of the FDDI token ring protocol are satisfied. *IEEE Transactions on Communications*, 35(6), June 1987.

[4] Paulo Veríssimo. Redundant media mechanisms for dependable communication in token-bus LANs. In *Proceedings of the 13th Local Computer Network Conference*, Minneapolis-USA, October 1988. IEEE.

[5] Gerard LeLann. Critical issues in distributed real-time computing. In *Proceedings of the Workshop on communication networks and distributed operating systems within the space environment*. European Space Research and Technology Centre, October 1989.

[6] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *ACM Transactions on Prog. Lang. and Systems*, 4(3), July 1982.

[7] Flaviu Cristian. Synchronous atomic broadcast for redundant broadcast channels. Technical report, IBM Almaden Research Center, 1989.

[8] R.Mangala Gorur and Alfred C. Weaver. Setting target rotation times in an IEEE Token Bus network. *IEEE Transactions on Industrial Electronics*, 35(3), August 1988.

[9] D. Dykeman and W. Bux. An investigation of the FDDI media-access control protocol. In *Proceedings of the EFOC/LAN Conference*, Basel, Switzerland, June 1987.

[10] Raj Jain. Performance analysis of FDDI token ring networks: effect of parameters and guidelines for setting TTRT. In *Proceedings of the ACM-SIGCOM'90 Symposium*, Philadelphia-USA, September 1990.

[11] C. Guerin, H. Raison, and P. Martin. Procedé de diffusion sûre de messages dans un anneau et dispositif permettant la mise en oeuvre du procedé. *French Patent*, (85.002.02), January 1985.

[12] P. Veríssimo, J. Rufino, and L. Rodrigues. Enforcing real-time behaviour of LAN-based protocols. In *Proceedings of the 10th IFAC Workshop on Distributed Computer Control Systems*, Semmering, Austria, September 1991. IFAC.

[13] J. Rufino and P. Veríssimo. A study on the inaccessibility characteristics of ISO 8802/4 Token-Bus LANs. In *Proceedings of the IEEE INFOCOM'92 Conference on Computer Communications*, Florence, Italy, May 1992. IEEE. also INESC AR 16-92.

[14] P. Veríssimo and José A. Marques. Reliable broadcast for fault-tolerance on local computer networks. In *Proceedings of the 9th Symposium on Reliable Distributed Systems*, Huntsville, Alabama-USA, October 1990. IEEE. Also as INESC AR/24-90.

[15] X3T9.5 FDDI. *FDDI documents: Media Access Layer, Physical and Medium Dependent Layer, Station Mgt.*, 1986.

[16] D. W. Andrews and G. Schultz. A token-ring architecture for local area networks. In *IEEE COMPCON*, October 1982.

[17] N. Strole. A local communications network based on interconnected token-access rings: A tutorial. *IBM J. Res. X Development*, 27(5), September 1983.

[18] Werner Bux. Local-area subnetworks: A performance comparison. *IEEE Transactions on Communications*, 29, October 1981.

[19] H. Okada, T. Yamamoto, Y. Nomura, and Y. Nakanishi. Comparative evaluation of Token-Ring and CSMA/CD medium access protocols in LAN configurations. In *Computer Networking Symposium*. IEEE, December 1984.

[20] J. Peden and A. Weaver. Performance of priorities on an 802.5 token ring. In *Proceedings of the SIGCOM'87 Symposium*, pages 58–66. ACM, 1987.

[21] Jay K. Strosnider and Thomas E. Marchok. Responsive, deterministic IEEE 802.5 token-ring scheduling. *Real-Time Systems*, 1(2), September 1989.

[22] J. Tusch, H. Meyr, and E. Zurfluh. Error handling performance of a Token Ring LAN. In *Proceedings of the 13th Conference on Local Computer Networks*, pages 355–363, Minneapolis, USA, October 1988. IEEE.

[23] *Token-Ring Adapter Chipset TMS380 User's Guide*, 1985.

[24] *ISO DP 8802/5-85, Token Ring Access Method*, 1985.