Availability of ARINC 629 Avionic Data Bus

Alban Gabillon
IUT de Mont de Marsan, Université de Pau
LIUPPA/CSySEC
371 rue du ruisseau, BP 201, 40000 Mont de Marsan – France
Email: alban.gabillon@univ-pau.fr

Laurent Gallon
IUT de Mont de Marsan, Université de Pau
LIUPPA/CSySEC
371 rue du ruisseau, BP 201, 40000 Mont de Marsan – France
Email: laurent.gallon@univ-pau.fr

Abstract— **The three traditional objectives of computer security are confidentiality, integrity and availability [8]. Availability can be defined as the prevention of denial of service. Confidentiality and integrity have been addressed in several theoretical works whereas the concept of availability has not been much investigated by the scientific community. This paper is an attempt to define through a case study the concept of availability. We first define a set of availability constraints that avionic data bus protocols should enforce. Then, we consider the ARINC 629 Basic Protocol (BP) and the ARINC Combined Protocol (CP) [2] which were implemented on the Boeing 777. We check whether these protocols respect our availability constraints or not.**

Index Terms—**availability, avionic data bus, denial of service, security**

I. Introduction

The three traditional objectives of computer security are confidentiality, integrity and availability [8]. Availability can be defined as the prevention of denial of service. A denial of service is an unauthorized withholding of information or resources. A denial of service can occur accidentally or it can take place as the consequence of a malicious action. Confidentiality and integrity have been addressed in several theoretical works whereas the concept of availability has not been much investigated by the scientific community. Let us, however, mention the models of Yu & Gligor [12], Millen [10] and Cuppens & Saurel [5].

As systems on Aircraft became progressively more digital in nature, it became apparent to avionic designers that a multiplexed bus system was required to enable all subsystems to be connected by only one set of wires. Since avionic systems place great emphasis on the reliable and timely transfer of information, specific data bus protocols for avionic systems have been designed. ARINC 629 is one of them. ARINC Specification 629 [2] defines a digital communication system, where Line Replaceable Units (LRUs : terminals connected to the network) may transmit and receive digital data using a standard protocol. A linear topology (bus) is used, and the protocol can be described as a Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA). The bus access control is distributed among all the participating terminals. ARINC 629 was implemented on Boeing 777. There exist two implementations of ARINC 629:

- Combined Mode Protocol (CP)
- Basic Mode Protocol (BP)

In [7], we made a timing analysis of the ARINC 629 CP by using Stochastic Timed Petri Nets [6,9]. In [13], we did the same for the BP. In [14], we contributed to the definition of the concept of availability through a case study. We first defined a set of availability constraints that Medium Access Control (MAC) protocols of avionic data buses should enforce. Then, we checked whether ARINC 629 BP conforms to these constraints or not. In this paper we extend the work presented in [14] by considering also the CP.

The remainder of this paper is organized as follows. Section II reviews previous works and informally presents our approach. Section III defines a language for specifying availability policies. This language is based on mathematical logic with deontic and temporal modalities. In section IV, we use our language to specify availability constraints that Medium Access Control (MAC) protocols should enforce. Section V gives an overview of ARINC 629 BP and CP. Section VI confronts the BP to our constraints. Section VII does the same for the CP. Section VIII analyzes the availability of ARINC 629 BP and CP. Section IX concludes this paper.

II. Approach and related works

Yu and Gligor [12] characterize availability as “how to provide a shared service with a specified *maximum*

Based on “Availability Constraints for Avionic Data Buses”, by Gabillon, A.; Gallon, L. [14] which appeared in the Proceedings of the First International Conference on Availability, Reliability and Security, ARES 2006, Vienna, Austria, April 2006.
This work has been funded by the French ministry of research under the label “ACI Sécurité Informatique 2004-2007”. Project METROSEC

waiting time". They use temporal logic to specify availability constraints. They also introduce the notion of *user agreements*. User agreements are constraints on the behavior of subjects. Yu and Gligor argue that lack of specifications for these agreements makes it impossible to demonstrate denial of service prevention.

Millen proposes a model [10] which resembles the Yu & Gligor model. In particular, the model includes the concept of *waiting time policy* defined by Yu and Gligor. The difference between the Yu and Gligor model and that of Millen's is that Millen uses a set-theoretic approach and includes an explicit representation of time.

Cuppens and Saurel [5] propose another approach. They define a logic-based language to specify availability policies. Their language basically includes two major predicates:

- *disposal_right*(*s*,*r*,*d*) reads subject *s* is permitted to access resource *r* and must be provided with *r* within a maximum waiting time of *d*.
- *use_right*(*s*,*r*,*d*) reads subject *s* is permitted to use resource *r* for a maximum duration of *d*.

With the first predicate, Cuppens & Saurel define the *waiting time policy*. With the second predicate, they define the *use time policy*. Holding a *disposal right* on a resource implies holding a *use right* on the same resource. Cuppens and Saurel express the meaning of both predicates in terms of deontic modalities. With their language, they also state some availability properties that the system should enforce. Basically, these properties express the fact that there is a security violation whenever the policy is not respected.

As we can see, Cuppens and Saurel declare explicit permissions in their policy. Consequently, there is a difference between the *waiting time policy* of Cuppens & Saurel and the *waiting time policy* of Millen and Yu & Gligor. For Millen and Yu & Gligor, the waiting time policy consists of a security constraint which says that a subject which asks for a resource has to be provided with it with a maximum waiting time. The policy of Millen and Yu & Gligor does not refer to explicit permissions.

Our approach resembles the Cuppens & Saurel model. In particular, the language we propose in section III is inspired from their language. Our approach can be summarized as follows: we do not define availability policies for avionic data buses. We rather state a set of availability constraints that MAC protocols of avionic data buses should enforce. Informally, we state that the access control policy of an avionic data bus should respect the following principles:

- Only one terminal at a time should hold the right to use the bus.
- There should be a fair distribution of rights among the terminals. Each terminal should be granted the right to use the bus once per bus cycle.
- If a terminal has been granted the permission to use the bus and actually uses it, then it should be permitted to use the bus as long as it needs, provided it does not exceed a specified duration.

We shall say that the bus is fully available if and only if these principles always apply. In the last part of this paper we study the conformance of the ARINC 629 BP and CP access control protocols with our availability constraints and we discuss the concept of denial of service.

III. LANGUAGE

In this section, we define the language which we use to express the availability constraints. This language can be seen as a simplified version of the language proposed by Cuppens & Saurel [5].

The language is based on first order logic with equality. It is extended with temporal logic and deontic concepts.

A. Temporal notions

The temporal dimension of the language comes from Sripada [11]. The language uses a discrete representation of time:

- $\mathrm{date(t)}$ reads *t* is a date.

The language introduces the following two temporal modalities:

If *p* is a formula and *t* is a date then,

- $\mathrm{p_t}$ reads *p* is true at time *t*
- $\mathrm{p_{t_1,t_2}}$ reads *p* is true during the temporal interval [t_1,t_2]

Axiomatic of these modalities is the following:

- $(\mathrm{p} \wedge \mathrm{q})_\mathrm{t} \leftrightarrow \mathrm{p_t} \wedge \mathrm{q_t}$
- $(\neg \mathrm{p})_\mathrm{t} \leftrightarrow \neg \mathrm{p_t}$
- $\mathrm{p_{t_1,t_2}} \leftrightarrow \forall \mathrm{t}, \mathrm{t_1} \leq \mathrm{t} \leq \mathrm{t_2} \rightarrow \mathrm{p_t}$
- if *p* is a theorem then p_t is also a theorem for every date *t*.

We shall use the following abbreviation:

- $\mathrm{always(p)} \leftrightarrow \forall \mathrm{t}, \mathrm{date(t)} \rightarrow \mathrm{p_t}$

Our language includes the following constant $\mathrm{D_{max}}$ which represents an arbitrary number of time units.

B. Deontic modalities

The language uses the following classical deontic modalities: **O**, **F** and **P** [4].

- **O**p reads *p* is mandatory
- **F**p reads *p* is forbidden
- **P**p reads *p* is permitted

We have the following equivalences:

- $\mathbf{F}\mathrm{p} \leftrightarrow \mathbf{O}\neg \mathrm{p}$
- $\mathbf{P}\mathrm{p} \leftrightarrow \neg \mathbf{F}\mathrm{p}$

Axiomatic of **O** is the following:

- $\mathbf{O}(\mathrm{p} \wedge \mathrm{q}) \leftrightarrow \mathbf{O}\mathrm{p} \wedge \mathbf{O}\mathrm{q}$
- $\neg(\mathbf{O}\mathrm{p} \wedge \mathbf{O}\neg \mathrm{p})$
- if *p* is a theorem then **O***p* is also a theorem.

C. Subjects

We use the following predicate to represent the subjects:

- $\mathrm{LRU(s)}$ reads *s* is an LRU (terminal)

Following constants represent the different terminals which are connected to the bus:

- $\mathrm{L_1}, \mathrm{L_2} \ldots \mathrm{L_n}$

We state the following axiom:

- $\text{always}\begin{pmatrix}\forall s, \text{LRU}(s) \\ \leftrightarrow s = L_1 \vee s = L_2 \vee \ldots \vee s = L_n\end{pmatrix}$

Following predicates respectively represent the fact that a subject uses the bus and the fact that a subject needs the bus:

- $\text{use}(s)$	reads *s* uses the bus
- $\text{need}(s)$	reads *s* needs the bus

We assume the following integrity constraints hold:

- $\text{always}(\forall s, \text{use}(s) \to \text{LRU}(s))$
- $\text{always}(\forall s, \text{use}(s) \to \text{need}(s))$

First constraint says that if *s* is using the bus then *s* is an LRU. Second constraint says that if *s* uses the bus then *s* needs the bus. In other words this constraint assumes that a subject would not use the bus if it does not need it. Such an integrity constraint is referred to as a *user agreement* by Yu and Gligor [12].

IV. Availability constraints for avionic buses

In this section, we state some availability constraints that MAC protocols of avionic data buses should enforce. Our proposal comes from our comprehension of what the "*ideal*" MAC layer of an avionic bus should be. Principles like fair distribution of rights and optimization of bus utilization have guided us for stating those constraints.

We consider that the ideal MAC protocol for avionic data buses should respect the following constraints:

The first constraint says that only one terminal at a time can be granted the permission to access the bus:

$$\text{always}(\forall s \forall s', \mathbf{P}\text{use}(s) \wedge \mathbf{P}\text{use}(s') \to s = s') \quad (1)$$

The second constraint says that, at any time, there is at least one terminal which is granted the permission to access the bus:

$$\text{always}(\exists s, \mathbf{P}\text{use}(s)) \quad (2)$$

From constraint 1 and constraint 2, we can easily deduce that there is always one and only one terminal which is granted the permission to access the bus.

The third constraint says that a terminal which was granted the right to use the bus cannot be granted the right to use the bus again before all the other terminals have been granted the right to use the bus. In other words, distribution of rights has to be fair:

$$\begin{aligned}&\forall s \forall t_1 \forall t_2 \forall t_3, [\mathbf{P}\text{use}(s)]_{t_1} \wedge [\mathbf{F}\text{use}(s)]_{t_2} \wedge [\mathbf{P}\text{use}(s)]_{t_3} \\ &\wedge (t_1 < t_2 < t_3) \qquad (3) \\ &\quad \to \begin{pmatrix}\forall s', \text{LRU}(s') \wedge s' \neq s \\ \to \exists t_4, [\mathbf{P}\text{use}(s')]_{t_4} \wedge (t_1 < t_4 < t_3)\end{pmatrix}\end{aligned}$$

The fourth constraint says that a terminal which is granted the right to access the bus holds the right as long as it needs the bus:

$$\forall s \forall t, [\mathbf{P}\text{use}(s)]_t \wedge [\text{need}(s)]_{t+1} \to [\mathbf{P}\text{use}(s)]_{t+1} \quad (4)$$

However, the fifth constraint says that the amount of consecutive time a terminal can hold the right to use the bus is bounded:

$$\forall s \forall t_1 t_2, [\mathbf{P}\text{use}(s)]_{t_1, t_2} \to t_2 - t_1 < D_{\max} \quad (5)$$

The last constraint says that if a terminal is permitted to use the bus and if it needs the bus then it is mandatory that the terminal uses the bus.

$$\text{always}(\forall s, \mathbf{P}\text{use}(s) \wedge \text{need}(s) \to \mathbf{O}\text{use}(s)) \quad (6)$$

We could easily show that these constraints are consistent, except in one case: if a terminal has been holding the right to use the bus for a duration which has become greater than D_{max} then constraint 4 conflicts with constraint 5. We solve this conflict by considering that constraint 5 is of a higher priority than constraint 4. Constraint 4 and constraint 5 define the *use time policy*.

We do not assign priorities to terminals. We claim that priorities are application dependent. Priorities, if any, have to be defined by the availability policy of an upper layer. Consequently, our security policy does not say in which order the terminals receive the right to use the bus. Our constraints do not express an explicit *waiting time policy*. However, from constraints 3 and 5, we can derive that the maximum amount of time a terminal has to wait before being granted with the permission to use the bus is equal to $(n-1) \times D_{\max}$ units of time. This maximum waiting time can theoretically be reached in the following situation:

- The terminal has just released the bus and is waiting for its next turn.
- Each of the *n*-1 other terminals uses the bus for the D_{max} duration.

The MAC layer of an avionic bus is secure if and only if the security policy is not violated. This means that the following two *security constraints* should be enforced:

The first security constraint says that a terminal which uses the bus should be permitted to use the bus. Violation of this rule is an *access control violation:*

$$\text{always}(\forall s, \text{use}(s) \to \mathbf{P}\text{use}(s)) \quad (A)$$

The second property says that if it is mandatory that a terminal uses the bus then the terminal should actually be using the bus. Violation of this rule is a *denial of service*:

$$\text{always}(\forall s, \mathbf{O}\text{use}(s) \to \text{use}(s)) \quad (B)$$

V. ARINC 629

A. *ARINC 629 Basic Protocol*

The ARINC 629 Basic Protocol is one implementation of the ARINC specification 629. The purpose of this section is to briefly describe the medium access control policy of this protocol. For more detailed information the reader can refer to [2] or [3].

An ARINC 629 BP message has variable length up to 31 wordstrings. Each wordstring is made up of one label and several data words. Each message contains the following types of data:

- Periodic data: each terminal regularly sends some periodic data at every bus cycle. For a given terminal, the length of the periodic data is generally fixed.
- Aperiodic data: occasionally, after a particular event, a terminal may also insert some aperiodic data in its message.

Bus time is decomposed into bus cycles. Each terminal sends one message per bus cycle. The ARINC 629 BP defines a Carrier-Sense Multiple Access/Collision Avoidance (CSMA/CA) scheme. Bus access control is distributed amongst all participating terminals, each of which autonomously determines its transmission sequence. This is achieved by the use of bus access timers as follows:

- Transmit Interval (TI): same for all terminals; 0.5 to 64.0ms; the longest timer; starts every time the terminal starts transmitting; is equal to the minimum bus cycle time.
- Synchronization Gap (SG): same for all terminals; values are 17.7µs, 33.7µs, 65.7µs, 127.7µs; ensures that all terminals are given access; chosen to be greater than the maximum TG; starts every time *Bus Quiet* (BQ) is sensed and may be reset before it has expired if bus activity sensed; restarted next time the terminal starts to transmit.
- Terminal Gap (TG): used to differentiate between terminals; unique for each terminal; values are 3.7µs to 127,7µs; starts only after the SG has expired and the bus is quiet; reset if bus activity sensed.

The initialization phase of bus operation may be relatively uncontrolled depending on the sequence in which local clocks are initialized across multiple terminals. However, as a result of synchronization mechanisms bus operation will normally settle down to some steady state after a very short time.

Figure 1 shows the BP periodic mode timing for a very simple bus with just three terminals. The order in which terminals achieve bus access is the same for all bus cycles and is determined by the initialization sequence. It does not necessarily equate to the relative durations of the terminals' TG timers. If the sum of all the TGs, transmission times and SG is less than TI then the bus cycle time remains fixed and equal to TI.

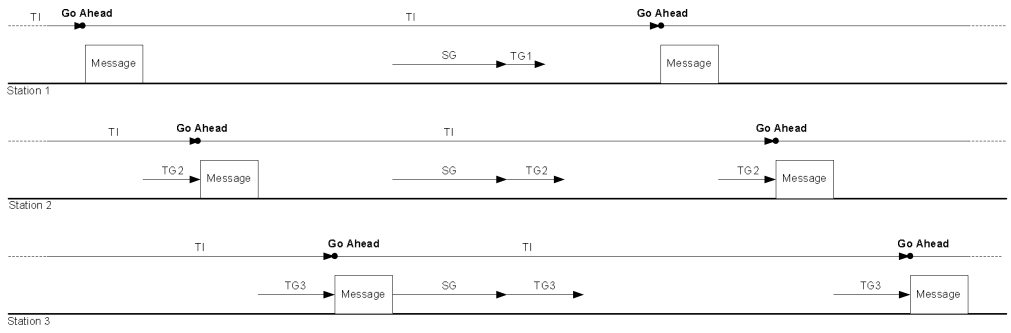

Figure 1 : BP periodic mode

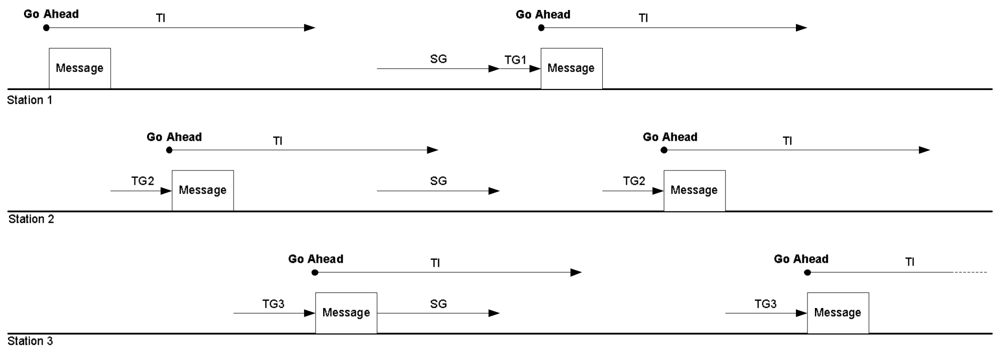

Figure 2 : BP aperiodic mode

If the sum of all the TGs, transmission times and SG is greater than TI then the system operates in aperiodic mode as shown in Figure 2. For every bus cycle the terminals transmit in the order of their TG durations. Here, TG1<TG2<TG3. Each cycle consists of a sequence of transmissions separated by the various TG delays; these are followed by an SG delay that provides synchronization. The two modes are not exclusive and the aperiodic mode will revert to periodic mode after transient overload. In fact, the system operates in the aperiodic mode when too many terminals insert aperiodic data in their message.

B. *ARINC 629 Combined Protocol*

The ARINC 629 Combined Protocol is the second implementation of the MAC layer of the ARINC specification 629. For more detailed information the reader can refer to [2] or [3].

The ARINC 629 CP defines a Carrier-Sense Multiple Access/Collision Avoidance (CSMA/CA) scheme. Whereas a BP messages may include perdiodic and aperiodic data, a CP message contains either periodic data or aperiodic data. Each bus cycle is divided into three main sections (see figure 3). The first section is called level 1, and serves for the periodic traffic. At level 1, each terminal must transmit one and only one periodic message. The second section is called level 2, and serves for urgent aperiodic traffic. The third section is called level 3, and serves for non urgent aperiodic traffic. This last level can be repeated if the bus cycle has not expired, and if some terminals still have non urgent aperiodic messages to send.

The Concatenation Event (CE) occurs at the beginning of the first periodic message. The terminal which has sent the first periodic message is called the leader. The CE gives the start of a new bus cycle for all terminals.

Bus access control is distributed amongst all participating terminals. This is achieved by the use of five bus access timers as follows:

- Transmit Interval (TI): same for all terminals; 0.5 to 64.0ms; the longest timer; starts on Concatenation Event (CE), or every time the terminal starts its periodic transmission (if this terminal is the leader); is equal to the bus cycle time. TI plays the same role as it does in the BP protocol.
- Aperiodic Access Time Out (AT) : same for all terminals, defines the last instant in the bus cycle for beginning an aperiodic transmission; AT = TI – (PSG + ASG + MAL), MAL being the Maximum Allowed Length for an aperiodic message; AT starts on Concatenation Event (CE), or every time the terminal starts its periodic transmission (if this terminal is the leader).
- Aperiodic Synchronization Gap (ASG): same for all terminals; values are 17.7µs, 33.7µs, 65.7µs, 127.7µs; Controls changes of level in bus cycles, chosen to be greater than the maximum TG, starts every time *Bus Quiet* (BQ) is sensed and may be reset before it has expired if bus activity sensed; restarted on every change of level.
- Periodic Synchronization Gap (PSG) : same for all terminals; PSG=5.ASG; controls changes of bus cycles; starts every time *Bus Quiet* (BQ) is sensed and may be reset before it has expired if bus activity sensed; restarted next time the terminal starts to transmit a periodic message; PSG plays the same role as SG does in the BP protocol.
- Terminal Gap (TG): used to differentiate between terminals; unique for each terminal; values are 3.7µs to 127,7µs; starts only after the PSG has expired and the bus is quiet; reset if bus activity sensed. TG plays the same role as it does in the BP protocol.

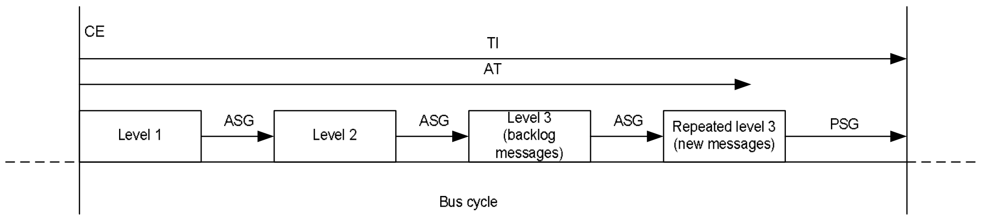

Figure 3: Bus cycle structure in CP

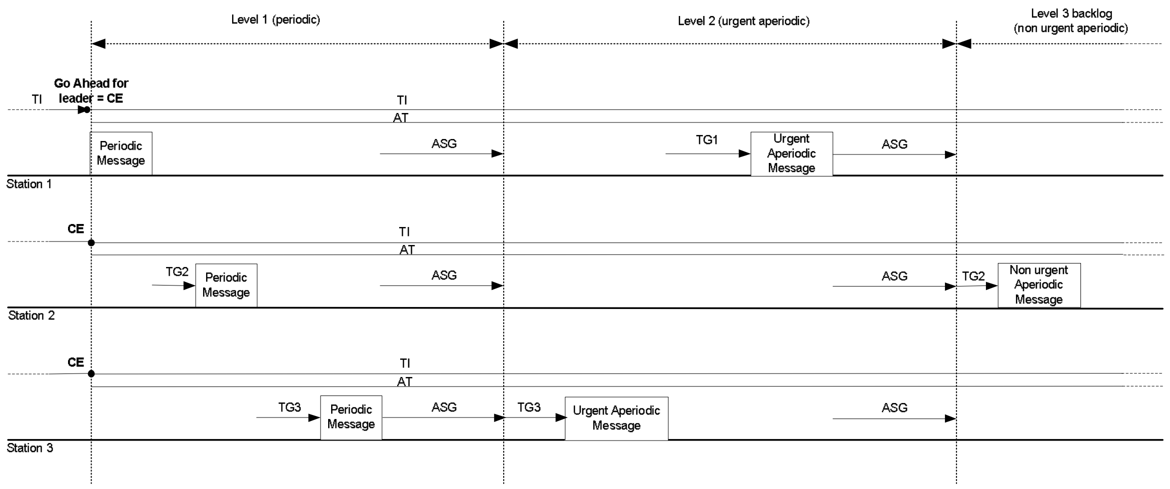

Fig ure 4: Bus cycle levels in CP

Figure 3 shows the structure of a bus cycle. At the end of each level, terminals run their ASG (synchronization) before accessing to the next level. Note that level 3 can repeat several times if the bus cycle has not expired. In fact, the first level 3 is reserved to the so-called backlog level 3 messages, i.e. level 3 messages which could not be transmitted in the previous bus cycle. New level 3 messages should be transmitted in other levels 3. When AT has expired, no new aperiodic transmission can start. All terminals have to run their PSG (synchronization) before accessing to a new bus cycle (TI expired for leader, CE for the others terminals).

Figure 4 focuses on the train of messages at the different levels. Except for the leader at level 1, terminals gain the right to transmit when their TG has expired. Each terminal should send a periodic message at level 1, but this is not mandatory at other levels. Terminals cannot send more than one message per level. Figure 4 shows all terminals sending one message at level 1, terminal 3 and terminal 1 sending one message at level 2, and, finally, terminal 2 sending one message at level 3 backlog.

The initialization phase of bus operation may be relatively uncontrolled depending on the sequence in which local clocks are initialized across multiple terminals. However, as a result of synchronization mechanisms bus operation will normally settle down to some steady state after a very short time. Note that the first terminal which starts transmitting (the leader) is not necessarily the terminal which has the smaller TG, but the first terminal which has been initialized.

VI. Analysis of ARINC 629 BP

In this section we check whether ARINC 629 BP conforms to the constraints we defined in section IV. Our purpose is not to make a formal analysis of this protocol. Our aim is to consider some particular configurations in order to understand the concept of availability and denial of service.

For our analysis, we choose the following four configurations:

- We consider the periodic mode after the initialization phase that is, after the bus has settled down to some steady state.
- We consider the aperiodic mode.
- We consider the initialization phase.
- We consider the transition between the periodic mode and the aperiodic mode

A. Periodic mode

Periodic mode is described by figure 1. A terminal is permitted to transmit (to use the bus) when it reaches the go-ahead condition. The terminal reaches that condition immediately after its three timers have expired. In figure 1, we can see that each terminal which reaches the go-ahead condition can actually start transmitting. Therefore, constraint 6 is satisfied. After it has started transmitting, a terminal is permitted to transmit as long as it needs and cannot be pre-empted. Therefore, constraint 4 is respected. A terminal cannot send a message which is more than 31 wordstrings (see section V sub-section A). Wordstrings are separated from each other by a 4 bits gap. Since the length of a wordstring ranges from 20 to 5140 bits and since a 2Mbps serial data transmission is specified for twisted pair conductor, the maximum transmission time of a message is 79.73 ms. Therefore, constraint 5 applies and constant D_{max} is equal to 79.73 ms. No terminal has its three timers expired while another terminal is still transmitting. Therefore, constraint 1 is satisfied. Within a single bus cycle, terminal 1 transmits then terminal 2, then terminal 3. The same procedure restarts at the next bus cycle. Therefore, constraint 3 is satisfied. In figure 1, we can see that after terminal 3 has released the bus none of the terminals have their timers expired yet. Therefore, none of the terminals have the permission to use the bus. Consequently **constraint 2 is violated.**

B. Aperiodic mode

Aperiodic mode is described by figure 2. If we proceed in the same way as we did for the periodic mode then we can easily show that only constraint 2 is violated. Indeed, there are some time intervals where all the terminals are waiting for some timers to expire. Other constraints are never violated.

C. Initialization phase (worst case)

We could show that during a safe initialization phase only constraint 2 is violated. We consider the worst case of an initialization phase which leads to a bus clash. Figure 5 describes this configuration. Terminal 2 starts before terminal 1. Then, both terminals run their SG and TG (constraint 2 is violated). After their SG and TG have expired, both terminals are permitted to transmit at the same time. Therefore, **constraint 1 is violated**. Both terminals try to transmit at the same time. A Bus Clash (BC) is detected. Both terminals enter into a recovery phase. **Security rule B is violated** since none of the terminals could use the bus whereas it was mandatory that both terminals could use the bus. After their TI has expired, terminals have to run their TG again. Since TG1expires first, terminal 1 is the first to transmit data.

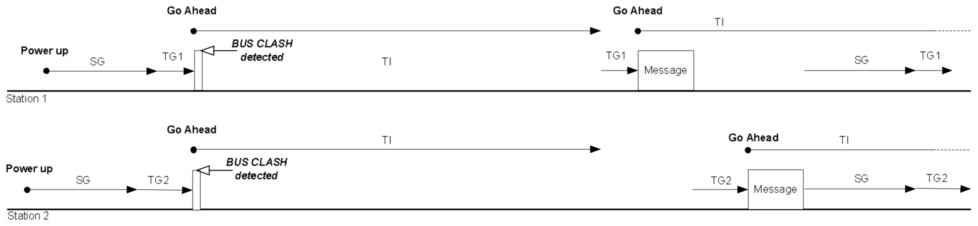

Figure 5 : Bus Clash

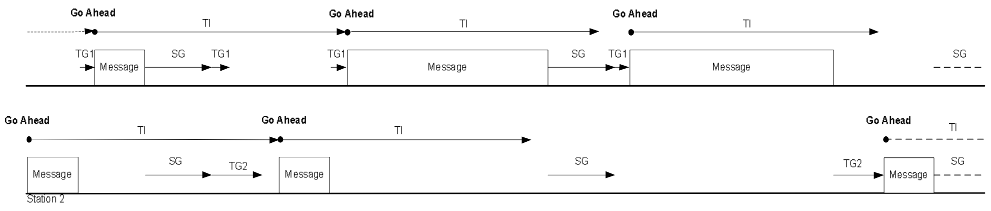

Figure 6: Transition between periodic mode and aperiodic mode

D. *Transition between modes*

Figure 6 shows a particular case of transition from periodic to aperiodic mode. At the end of the initialization phase, terminal 2 is the first to transmit data. During the first cycle the bus is operating in the periodic mode. The order in which the terminals transmit depends on bus initialization. At the second bus cycle the order in which the terminals send their message remains the same (first terminal 2 then terminal 1). However, since terminal 1 has sent a large message, TI timers have expired before the SG timers. During this second cycle, the bus is operating in a mode which can be seen as a transient mode between the periodic and the aperiodic mode. At the third cycle, the bus is operating in the aperiodic mode. The order in which the terminals send their messages has changed. The terminals now transmit in the order of shortest to longest TG. Terminal 1 is the first to transmit data then terminal 2. To sum up, the order in which the messages are sent is the following (m_it_j denotes the i^{th} message of terminal *j*):

- m_1t_2, m_1t_1, m_2t_2, $\mathbf{m_2t_1}$, $\mathbf{m_3t_1}$, m_3t_2

Between the second bus cycle and the third bus cycle, **constraint 3 was violated** since terminal 1 was granted two times consecutively the permission to transmit. Distribution of rights was temporarily unfair.

VII. ANALYSIS OF ARINC 629 CP

Let us first mention that a bus clash might occur during the initialization phase. We made the analysis of this case in section VI for the BP. The analysis we could make for the CP would be exactly the same and would provide us with the same results.

For analyzing the CP, considering figure 4 is indeed sufficient. However, we shall conduct our analysis from the following perspectives:

- We focus on each level.
- We consider the bus cycle.

A. *Analysis of level 1*

A terminal is permitted to transmit (to use the bus) when it reaches the go-ahead condition (if it is the leader) or after its TG has expired (if it is not the leader). The leader reaches the go-ahead condition immediately after its TI has expired. In figure 4, we can see that the leader which reaches the go-ahead condition can actually start transmitting. We can also see that other terminals which have their TG run out can also start transmitting. Therefore, constraint 6 is satisfied. After it has started transmitting, a terminal is permitted to transmit as long as it needs and cannot be pre-empted. Therefore, constraint 4 is respected. A terminal cannot send a message which is more than 31 wordstrings. Therefore, constraint 5 applies. No terminal has its TG expired while another terminal is still transmitting. Therefore, constraint 1 is satisfied. Within a single bus cycle, terminal 1 (the leader) transmits then terminal 2, then terminal 3. The same procedure restarts at the next level 1 (during the next bus cycle). Therefore, constraint 3 is satisfied. In figure 4, we can see that after terminal 3 has released the bus none of the terminals have their timers expired yet. Therefore, none of the terminals have the permission to use the bus. Consequently **constraint 2 is violated.**

B. *Analysis of level 2*

Level 2 works like level 1. The only minor differences are the followings:

- A terminal (including the leader) is permitted to transmit after its TG has expired. In figure 4, we do not see terminal 2 transmitting simply because it does not have any urgent aperiodic message to send (if it had one then it could have sent it before terminal 3).
- A terminal cannot send a message which is more than 1 wordstring. Therefore, constant D_{max} is equal to 2.57 ms.

Despite these small differences, the analysis of level 2 gives exactly the same results as analysis of level 1 i.e. only constraint 2 is violated.

C. *Analysis of level 3*

Level 3 works like level 2. However, the fact that it is the last level in the bus cycle may lead to the **violation of constraint 3**. Indeed, as soon as the AT timer expires, the protocol first waits for the end of the current aperiodic message, then forbids any kind of aperiodic transmission. Therefore, it is possible to have a situation where terminals with long TG could not have their TG run out and, consequently, did not receive the permission to transmit their aperiodic data. These terminals should wait for the next bus cycle and send their aperiodic data during the first level 3 i.e. the backlog level 3.

D. *Analysis of the bus cycle*

If we consider the bus cycle globally then we can identify other violations of constraint 3. At level 2 (and level 3), terminals receive the permission to transmit in the order of their TG, from the shortest to the longest. This is not the case at level 1. Indeed, at level 1, the leader transmits first, then the other terminals (in the order of their TG). If the leader is not the terminal with the shortest TG then constraint 3 is violated at level 2, since the leader is not the first terminal to transmit at that level. In figure 4, we can see that the leader is the terminal with the longest TG. It is the first to transmit at level 1 and the last to transmit at level 2. Between these two messages, terminal 3 could send two messages (one periodic and one aperiodic). A similar situation may occur from level 3 to the level 1 of the next bus cycle.

VIII. Availability of the ARINC 629 BP and CP

The BP and CP protocols enforce an access control policy which does not respect the availability constraints we define in section IV in the following cases:

- Constraint 2 is regularly violated. Indeed, within each bus cycle, there are time intervals when nobody has the right to transmit. During these intervals all the terminals are waiting for some timers to expire.
- Constraint 1 is violated in one rare collision case which might happen during the initialization phase. Violation of this constraint leads to a bus clash and a violation of security rule B.
- Regarding the BP only, if the bus is operating in the periodic mode and the terminals are not transmitting in the order of shortest to longest TG then constraint 3 is temporarily violated when the bus switches to the aperiodic mode.
- Regarding the CP only, if the leader is not the terminal with the shortest TG then constraint 3 is violated whenever the protocol switches from level 1 to level 2 and from the last level 3 to the level 1 at the next bus cycle. Constraint 3 can also be violated at level 3 if too many terminals need to send some aperiodic data. In that case, terminals with long TG may not receive the permission to send their aperiodic data.

As we can see, only violation of constraint 1 leads to a denial of service since only violation of constraint 1 leads to a violation of security rule B. Violation of constraints 2 and 3 leads to security violations only if we assume that the availability constraints defined in section IV are *mandatory* and *should* apply to the availability policy. In other words if the bus protocol has the *obligation* to enforce an access control policy respecting the availability constraints we define in section IV then violation of constraint 2 or constraint 3 leads also to security violations. Violation of constraint 2 leads to a denial of service. Let us consider a date which belongs to a time interval where all the terminals are waiting for some timers to expire. Since constraint 2 says that there should always be a terminal with the permission to use the bus, we can say that there is a denial of service for one of the terminals at this date. Violation of constraint 3 leads both to a denial of service and an access control violation. Indeed, example of section VI (sub-section D) shows that terminal 1 uses the bus two times consecutively. There are two security violations when terminal 1 uses the bus for the second time:

- Terminal 2 is victim of a denial of service since, according to constraint 3, it should be its turn to use the bus.
- Terminal 1 illegitimately uses the bus since, according to constraint 3, it should not be its turn to use the bus.

It is, however, important to notice that if we consider only periodic data then the CP never violates constraint 3. Indeed from level 1 to other level 1s, the order in which terminals transmit remains the same (first the leader, then the other terminals in the order of their TG). In this consideration lies the main practical difference between the BP and the CP. The BP is simpler than the CP since it uses only three timers. However, the combined protocol offers more stable periodic response than the basic protocol for systems which require some combination of periodic and sporadic message transmissions since there is no potential for aperiodic messages to interfere with periodics.

On the contrary, the extent to which required aperiodic behaviour can be guaranteed under the CP depends on the degree of acceptable under-utilisation of the databus (violation of constraint 2). Indeed if values assigned to timers TI and AT are not large enough, then terminals with long TG may experience delays before being able to send their aperiodic data (violation of constraint 3).

In [1], Ausley and Grigg made a timing analysis of the ARINC 629. Their aim was to evaluate the ability of the ARINC 629 to support real-time applications. They proposed an approach to predict delays associated with worst case behavior. It is interesting to note that some of the worst case delays they identified occur when our availability constraints are violated.

IX. Conclusion

In this paper, we revisited the ARINC 629 from availability constraint standpoint. Our work is a contribution toward the definition of a general availability model. It could be extended in several directions:

We could extend our approach to the global problem of resource allocation.

We could use the same approach to analyze other protocols like Ethernet (which obeys a CSMA/CD philosophy).

We could investigate the problem of merging several availability policies. For example we could choose an avionic application and express its availability requirements. Then, we could merge the availability policy of the MAC layer with the availability policy of the application layer and see if conflicts appear. We feel that such approach would allow us to make the same analysis as Grigg and Ausley [1] made but in a more formal and complete way.

References

[1] Neil C. Ausley, Alan Grigg. *Timing analysis of the ARINC 629 data bus for real-time applications*. Microprocesssors and Microsystems 1997.

[2] Airlines Electronic Engineering Comittee. *Multi-Transmitter Data Bus. Part 1 : technical description*, Aeronautical Radio Inc. edition, march 1999

[3] B. Beizer, *The architecture and Engineering of digital computer complexes*, vol. 1, Plenum Press, 1971.

[4] B. Chellas. *Modal Logic: an introduction*. Cambridge University Press.

[5] F. Cuppens and C. Saurel, *Towards a formalization of availability and denial of service*. Information Systems Technology Panel Symposium on Protecting Nato Information Systems in the 21st century. Washington, 1999

[6] G. Juanole and L. Gallon, *Critical Time Distributed Systems: Qualitative and Quantitative Analysis Based on Stochastic Timed Petri Nets*, Formal Description Techniques VIII, Proceedings of the IFIP TC6 Eighth International Conference on Formal Description Techniques, Montreal, Canada, October 1995

[7] L. Gallon, G. Juanole and I. Blum, *Modelling and Analysis of the ARINC specification 629 CP Mac Layer protocol*, IEEE workshop on factory communication systems. Barcelona, 1997

[8] Information Technology Security Evaluation Criteria. June 1991.

[9] G. Juanole and L. Gallon, *Concept of Quantified Abstract Quotient Automaton and its Advantage*. IFIP TC6 WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE X), 18-21 November, 1997, Osaka, Japan

[10] J.K. Millen. *A resource allocation model for denial of service*. IEEE Transactions on Security and Privacy, Oakland, USA 1992.

[11] S.M. Sripada. *A metalogic programming approach to reasoning about time in knowledge bases*. Proceedings of the 13th International Joint Conference on Artificial Intelligence.

[12] C. Yu and V. Gligor. *A specification and verification method for prevention of denial of service*. IEEE Transactions on Software Engineering 16(6): 581-592, June 1990.

[13] A. Gabillon, L. Gallon *An Availability Model for Avionic Databuses***.** Proceedings of the Workshop on Issues in Security and Petri Nets (WISP). Eindhoven, The Netherlands, 23 June 2003.

[14] A. Gabillon, L. Gallon. *Availability Constraints for Avionic Data Buses*. Proceedings of the First International Conference on Availability, Reliability and Security, ARES 2006, Vienna, Austria, April 2006.

Alban Gabillon is a full Professor at the University of Pau in France. Alban Gabillon received his Ph.D. in Computer Science from the *Ecole Nationale Supérieure de l'Aéronautique et de l'Espace* in 1995. He has been working for more than 10 years on various topics of computer security including multilevel security, access controls to database, availability models and electronic time-stamping. He has published around 40 technical papers on computer security in refereed journals and conference proceedings. He is now in charge of the **C**onnected **Sy**stems **SEC**urity (LIUPPA/CSySEC) research group who deals with different research activities in the domain of Computer Security.

Laurent Gallon is Assistant Professor at the University of Pau in France. Laurent Gallon received his Ph.D. in Computer Science from the *Université Paul Sabatier* of Toulouse, in 1997. He has been working on Petri Nets. He is now member of the **C**onnected **S**ystems **SEC**urity (LIUPPA/CSySEC) research group who deals with different research activities in the domain of Computer Security. His research interests include computer security, communication network and formal specification methods.